
CALL MSX
Número 2 · 6 Euros
Reuniones MSX
SCREEN 2 a fondo
Dragon Slayer IV
BOMBA

Call MSX, número 2 Mayo 2004

índice

Call MSX son: Francisco Álvarez "Saeba" (saeba@iespana.es) - Óscar Centelles "Zanac 2" (kaneda@filnet.es) - Julio Gracia "MesíasMSX" (mesiasmsx@iespana.es) - David Lucena "Yakumo" (yakumoklesk@yahoo.es) - Iván Priego (ivan@moai-tech.com) - Eva Molina "Hyrule" (hyruleva@hotmail.com) - Pol Roca (pol.roca@campus.uab.es) - Roberto Álvarez (robertoat@iespana.es)

editorial

Como lo prometido es deuda, volvemos a encontrarnos varios meses después,
coincidiendo con la XXV Reunión de Usuarios de MSX en Barcelona. A pesar de las dificultades que conlleva coordinar el desarrollo de una revista cuando los integrantes de las mismas tienen sus propias obligaciones, os traemos un nuevo número de Call MSX que esperemos cubra vuestras espectativas. Hemos intentado
mejorar muchos de los aspectos en que habíamos flojeado en nuestro primer número, sobre todo en el tema de maquetación. Hemos introducido secciones nuevas que quizá alguno echaba en falta, y que esperemos que se conviertan en constantes dentro de la propia revista.
Uno de los apartados introducidos es el de programación. En éste se pretenderá dar una visión de los diferentes aspectos de la programación del MSX desde un punto de vista amateur.
Destacamos como en el anterior número, las novedades aparecidas recientemente, como ha sido el juego Bombaman, del grupo holandés Team Bomba. Sería deseable seguir contando
con ellas en los sucesivos números de call MSX, aunque siempre soñando con que el número de ellas sea mayor con el tiempo.
Seguimos intentando ofrecer una mayor variedad de temas, además de conservar también otros apartados que creemos de interés como el de Konamiteca, donde cada número repasaremos un éxito de Konami, la empresa que más vida ha dado al MSX en cuestión de videojuegos.
No nos olvidamos también de las guías sobre juegos, mapas, trucos y otros artículos que esperemos os hagan pasar un rato entretenido y agradable. Entre ellos no podemos olvidar los encuentros entre usuarios que son, en definitiva, importantísimos para que muchos de nosotros sigamos activos.
Quedaría comentaros que para el próximo número intentaremos seguir mejorando y por ello ya tenemos proyectados nuevos y muy interesantes artículos. Así, os convocamos para nuestra próxima aparición, deseando que volváis a confiar en nosotros, y por nuestra parte intentaremos devolveros el apoyo con alguna sorpresa y con todo aquello que un usuario pueda esperar de una revista dedicada, sin duda, a vuestro sistema favorito: el MSX.

El equipo de Call MSX.



24ª Reunión de Usuarios de MSX

Muchas líneas de código y comentarios diversos en su mayoría poco alentadores han sido vertidos a raíz del último encuentro de usuarios de MSX organizado por la AAM esta vez en conjunto con los responsables de la BCN party, que se celebró el Sábado día 1 de Noviembre de 2003.

Uno de los temas más comentados y sobre el que ha llovido más críticas ha sido el tema del día ce la celebración del encuentro. Mucha gente está acostumbrada a que el encuentro se celebre el día 8 de Diciembre aproximadamente, puesto que suele coincidir con el puente de la Constitución y el día de la Inmaculada Concepción. Y esto realmente sí es cierto que afecta a mucha gente, sobre todo la que tiene que trabajar, pero más aún si esta gente viene de fuera. Y dio la casualidad de que gente que no es de Barcelona, como Fernando López (a.k.a. Sapphire) o Rafael Corrales (a.k.a. MadriSX) asistieron al encuentro sin problema alguno. De todas maneras, todos aquellos que no asistieron podría aceptársele la excusa del puente como válida, aunque es bien cierto que faltó mucha más gente de la propia Cataluña que de las afueras.

Por otra parte se ha atacado el hecho de celebrar la reunión conjuntamente con la BCN Party. Seamos francos. Si ese mismo día la reunión se hubiera celebrado en las Cocheras de Sants, apenas a unos pares o tres de centenares de metros del Casal Hostafrancs, seguramente hubiera venido exactamente la misma cantidad de personas al encuentro, porque tal y como se repitió diversas veces, la Party y la RU quedaban mutuamente excluidas dentro del propio edificio. Cada una de ellas tenía un local totalmente independiente y con la total libertad para usarlo del modo que más creyeran conveniente, y exactamente así se hizo.

Por último se ha hablado de que mucha gente no se ha hecho con la noticia de la celebración del encuentro y culpan de ello al olvido de la AAMSX de enviar los correspondientes correos electrónicos a la gente que está en sus bases de datos. Ciertamente, y por motivos que no vienen al caso, hubo un desliz por parte de los miembros de la Asociación por los que el envío de los correos electrónicos avisando del encuentro no llegó a hacerse efectivo. Pero realmente poner esta excusa como motivo de la ausencia de la gente me parece realmente lamentable. Si la gente que uno espera hallar en una reunión únicamente va a aparecer porque sean avisadas del encuentro, y no porque tengan el suficiente interés en el MSX para buscar por ellos mismos la fecha de tal acontecimiento, es preferible no encontrar ninguna persona. Creo que la mayoría de usuarios de MSX, más o menos activos, somos lo suficientemente maduros como para, si realmente nos interesa el sistema, hacer el mínimo esfuerzo de acceder eventualmente a la página oficial de la asociación, localizada en www.aamsx.com, en donde se puede encontrar con suficiente antelación información sobre futuros encuentros, tanto de fechas como de stands asistentes y sus contenidos, precios de entradas, localización, etc. Y si por casualidad uno no dispusiera de internet, algo que casi ya podría considerarse preocupante en la época que estamos en la que la información ha tomado un papel tan importante que se ha convertido en principal moneda de cambio de la gran parte de las empresas, estoy seguro de que conoce a un amigo, o a un amigo de un amigo que puede hacerle el favor de decirle cuándo se va a producir el encuentro. Y si no también puede ir a un cibercafé. En definitiva, aquella persona que no sabe dónde ni cuando se realiza un evento de tales características, es porque no quiere saberlo o no le interesa lo suficiente.

Bien, tras esta versión personal que yo, David Lucena, he dado del encuentro, y que ha tenido como objetivo no solo rebatir algunas quejas sino dar una pequeña reprimenda a aquellos que se han quejado desde mi punto de vista sin razón, expondré lo que ha sido la reunión, hablando como siempre de los stands y sus contenidos, amén de algún que otro evento producido durante el transcurso del encuentro.

Taburoto

Estuvieron este grupo de usuario nuevamente en otro encuentro haciéndose escuchar, y a todo volumen, mediante la reproducción de su último CD recopilatorio de músicas hasta el momento denominado MSX 20th Anniversary, cuyas músicas en formato de Audio CD procedentes de músicas Midi tenían una calidad realmente buena, cuyos miembros se











encargaron de demostrar sobre todo en diversos momentos de la reunión. Esperemos que las personas que aún estuvieran durmiendo en las salas contiguas no se lo tomaran a mal. Para animar un poco más la reunión, a fin de que no quedara un poco soso el contenido del stand, el grupo tuvo la idea de poner diversas demos, algunas musicales y algunas más gráficas. Unas de ellas era una demo musical con las imágenes de todas las chicas aparecidas en el Cobra Mission, por lo que no hay que decir que sólo era apta para adultos. Además, Juan de Dios, nos mostró una pantalla de un juego al estilo del Alien Breed realizada con el editor Dante 2, mientras que un amigo suyo que hizo su aparición por primera vez en un encuentro de estas características estuvo durante un tiempo tocando con un estilo considerable el piano que allí se encontraba en un determinado momento de la reunión.

SD-Mesxes

Realmente se me hace extraño hablar de este stand. De hecho, si os soy sincero, incluso momentáneamente me lo había saltado porque no había caído en él. Y es que este stand no fue en esta reunión lo que solía ser en otros tiempos, puesto que únicamente contó con unos de sus miembros, Néstor Soriano (a.k.a. Konamiman), puesto que el resto de integrantes del grupo: Marce, Saver, Mato, Ramoni, no pudieron asistir al encuentro por motivos que escapan a mi conocimiento. Además, el hecho de que el último fanzine saliera a marchas forzadas y en el último día, auguraba que en este próximo encuentro no hubiera novedad alguna. Eso sí, los que se hubieran perdido algún número anterior podían adquirirlo en el mismo stand, además de algún que otro Internestor suite. También se podía jugar a algún que otro juego trasteando en los interiores del compact flash del TR que allí se encontraba, como ya hizo Juan Manuel de la Cruz con el Space Manbow, cuyo final pudo mostrarnos tras un intenso esfuerzo.
Lamentablemente no se pudo realizar la demostración del Internestor Lite que se tenía preparada al parecer debido a problemas con el driver Fossil modificado por Roberto Vargas.

Desgalitxat

Esta vez no estuvo Ángel Carmona al frente de uno de los stands que más tamaño suelen abarcar, pero desde luego el resto de componentes del grupo nos supo ofrecer con el máximo número de detalles todos los productos que allí se ofrecían. Se pudo destacar unas camisetas de estilo camuflaje realmente atractivas, con portadas del Metal Gear, tanto de MSX como de PSX. Además de esto nos trajeron unas nuevas camisetas con detalles fosforescentes que se iluminan en la oscuridad, y una de ellas incluía la portada del Maze Of Galious. Junto a estas ya agradecidas novedades podíamos contar con los habituales cuadros enmarcados de portadas de juegos así como el fabuloso Collection Covers en versión DVD con una carátula bastante conseguida.

AAMSX

Este stand, cuya presencia es imposible de que no exista, pues entonces no sería una Reunión de Usuarios de Barcelona, nos trajo una vez más juegos y hardware variado de segunda mano listos para ser vendidos. Y ciertamente que se vendió material. Sin ir más lejos se me escapó de las manos un Golvellius a un precio muy aceptable, lo que espero que no vuelva a pasar en próximos eventos. Se vendieron diversos Music Modules, uno de ellos ampliado, algún FM PAC e incluso un 8280 sin teclado, y con la carcasa algo deteriorada, pero en perfecto funcionamiento a un precio considerado como ganga en los tiempos que estamos viviendo. Además de esto existían dos ordenadores con sendos joysticks Telemach para que la gente fuera practicando sus habilidades en los juegos que iban a ser usados para los concursos que más tarde llegaron. Esta vez faltó la nota musical a la que normalmente nos tiene acostumbrado Jordi Tor (a.k.a. Melenas), integrante de la AAMSX, puesto que no trajo el equipo de audio que suele montar en cada reunión. Sin embargo si puso a la venta sus 2 CDs de músicas Midi en formato AudioCD y conocidos con el nombre de Perfect Covers, los cuales acabaron desapareciendo completamente.

Call MSX

Me siento muy orgulloso de llegar a este punto del comentario, ya que por fin llega el momento de hablar de nuestro stand. No es que existieran muchas novedades, salvo la propia revista, pero al ser nuestra primera incursión, al menos la mía, detrás de un stand como integrante de un equipo en lugar de estar en frente como un usuario corriente, la emoción









sentida es realmente intensa y agradable. Al principio hubo un poco de escepticismo, puesto que las ventas incrementaban a un ritmo verdaderamente lento, pero más tarde la cosa cambió, y pese que el número de ventas alcanzado no llegó a lo que se hubiera querido, éste se ha conseguido aumentar gracias al interés que muchas personas, incluso desde fuera de España, han mostrado por esta nueva publicación mediante la petición del envío por correo de ésta a su domicilio. Desde aquí agradecerles a todos por parte del equipo su confianza, y esperamos que podamos contar con ellos en sucesivos números. Durante el transcurso de la mañana, apareció por el lugar Juan Luis (a.k.a MSX-Kun), que trajo algunas copias de sus juegos, unas para ser entregadas a quienes se lo pidieron, yo entre ellos, y otras para ser vendidas en el propio encuentro, junto con las que regalaba un folletín o minizine pequeñito pero con noticias muy frescas e íntegramente realizado por él, con un aire un tanto cómico sin llegar al nivel sectario de la Mesxes. Nos pidió el favor de poder vender sus juegos en el propio stand, y puesto que había espacio de sobras entre montón de call MSX y call MSX y el chico la verdad es que no gana nada con ello puesto que el juego es gratuito, le dispusimos un espacio para que pudiera vender sus juegos en su carcasa de plástico, con manuales y portadas en color. Realmente merece la pena pagar dos euros por una presentación tan maja. De hecho, las copias volaron en muy poco tiempo, cuando aún no se había vendido prácticamente ninguna call MSX, lo que en parte me produjo un amago de depresión 8) que más tarde se eliminaría al ver como la gente aceptaba nuestra publicación haciéndose con ejemplares de la misma.

MOAI-TECH

Esta vez MOAI-TECH no nos trajo un nuevo número, ni tampoco estuvieron todos los integrantes del grupo, pero sí pudimos hacernos con números atrasados que allí estaban expuestos para su venta, atendida por los dos únicos miembros integrantes presentes, Julio Gracia (a.k.a. MesiasMSX) e Iván Priego, que ahora forman parte también de esta publicación.

Retro Gamez

A pesar de no aparecer en la página de la AAMSX, tuvimos la presencia de otro stand extra, que aparecieron aproximadamente una hora o dos después de comenzada la reunión. Estaba formado por un chico que espero que me perdone por no conocer su nombre, pero no se me pasó por la cabeza el preguntárselo. Trajo consigo una gran cantidad de juegos de segunda mano, tanto japoneses como europeos, algunos en mejor estado que otros, pero la mayoría sin demasiado desgaste, además de algunos juegos de otros sistemas como Super Nes o Gameboy. A medida que el tiempo iba pasando el poseedor del stand rebajaba los precios de



Diverso material en el stand de segunda mano

algunos juegos, ya que hay que decir que el precio inicial de muchos de ellos era un tanto elevado. De todas maneras, aquél que realmente tuviera interés en algún juego concreto siempre podía llegar a un acuerdo, dependiendo de lo buen comprador que fuera.

Los concursos

A lo largo del encuentro hubo dos concursos, que se desarrollaron por la mañana y por la tarde. El primero de ellos consistió en una liga eliminatoria al *Super Runner*, con una única partida por eliminatoria, en la que dos contrincantes tenían que competir por ser el mejor, llegar a la meta antes que el otro y a la vez, si era posible, conseguir la mayor cantidad de puntos posibles. Un servidor lamentablemente quiso intentarlo sin haber jugado en la vida a ese juego, y se enfren-










tó en su primera eliminatoria nada más y nada menos que a David Fernández. Sobra decir que ni tan sólo llegué a la mitad del recorrido, pues no sabía que se iba perdiendo energía, ni tampoco sabía qué era bueno o malo y ni mucho menos cómo coger el balón o huevo que era necesario llevar hasta al final. Sin embargo, y frente a todo pronóstico, no fue David



Jugando al Super Runner en el consurso

Fernández quien obtuvo el premio, si no Jaume Martí, lo que creo que fue aún más justo, puesto que de esta manera no siempre es el mismo quien se lleva los regalos, y además que lo consiga una persona que ha formado parte de una publicación que nos ha dado tantos ratos de entretenimiento (a pesar de que el que os escribe la conoció muy tarde y por casualidad en el sótano de Norma Cómics de Paseo san Juan), resulta más satisfactorio. Este primer premio consistió en una controladora FDD Brasileña y no Coreana como se informó en la página de la Asociación, pero el origen de esta no es lo que importa, si no el uso que se puede hacer de ella.

El segundo concurso se desarrolló tras el descanso del mediodía, y fue si cabe más emocionante que el primero, puesto que consistía en ganar en un combate al contrario en el juego Konami's Boxing. Esta vez llegué a semifinales, a pesar de no haber jugado gran tiempo a ese juego y sólo haber practicado los minutos anteriores al concurso. Pero el truco consistía básicamente a aporrear lo más rápido que pudieras el botón de disparo cuando pillaras al otro con la guardia baja, y de vez en cuando hacer algún que otro amago. Sin embargo, no por ello pude superar a Iván Priego, miembro también de esta revista quien al final resultó vencedor del concurso. Y estoy seguro de que se llevó una gran alegría y sorpresa al recibir el premio, puesto que no se esperaba que este fuera un Princess Maker en su caja, plastificado y con una apariencia completamente nueva, en un estado insuperable y que daba goce sólo de verlo. Creo que si me hubiera tocado a mí no me hubiera atrevido ni tan sólo a desenvolverlo.

Esto es lo que dio de sí la reunión. Luego más tarde, a la hora habitual asistimos a la cena los que siempre solemos ir, y que se celebró para no variar en el "Chino de la muerte". Lamentablemente yo me encontraba en un estado de gripe intestinal y solamente pude digerir y a duras penas un rollito de primavera, con lo que fue el rollito más caro que jamás hube probado. Destacar que a la cena acudieron Fernando López y Rafael Corrales, además de @@@@@ (a.k.a. Xenon Soft) quien no sólo apareció por el encuentro sino que además estuvo en calidad de representante del grupo de GP32Spain en un stand en el piso de abajo en la sala donde se desarrollaba la BCN Party. Para el que pueda estar interesado, decir que trajeron varias GP32 que vendían al precio de 200 euros, del modelo con retroiluminación, por cuya compra regalaban un CD con una colección de emuladores, juegos y programas bastante numerosa. Además de algunas tarjetas de almacenamiento y ciertos periféricos para algún que otro sistema. Según datos de Rafael Corrales, tuvieron un éxito de ventas bastante aceptable.

Pues bien, esto es todo lo que se puede contar del encuentro, y alguna anécdota más que me la guardo para mí y el resto de los que fuimos en conjunto al encuentro. Esperemos que podamos vernos en el siguiente con el mismo ánimo y ganas que en este, y que continuemos divirtiéndonos y disfrutando tanto del sistema como del propio encuentro en sí. Un saludo.

David Lucena





BOMBAMAN - Quest on Iko Iko Island

Las novedades del estándar se ausentan cada vez más, año tras año. Es una pena que juegos avanzados en su desarrollo no se lleguen a concluir por uno u otro motivo (no hace falta recordar los casos) o que otros se alarguen en el tiempo sin llegar a ver la luz. Éste se podría haber englobado en el segundo grupo, pero el afán por terminarlo ha podido más y, por fin, tenemos entre nuestras manos la última producción holandesa. Ya dio que hablar hace unos añitos, con la



inclusión de una demo en el disco de la entrada de la vigésima Reunión de Usuarios de MSX de Barcelona. Ahora, poco más de un año después de aquella primera toma de contacto, Bombaman ya puede ser adquirido a través de Sunrise al módico precio de 14 euros desde su aparición en el encuentro holandés de Oss. La distribución para España, como de costumbre, corre a cargo de Hnostar. El precio para los usuarios españoles es de 16 euros.

EL JUEGO

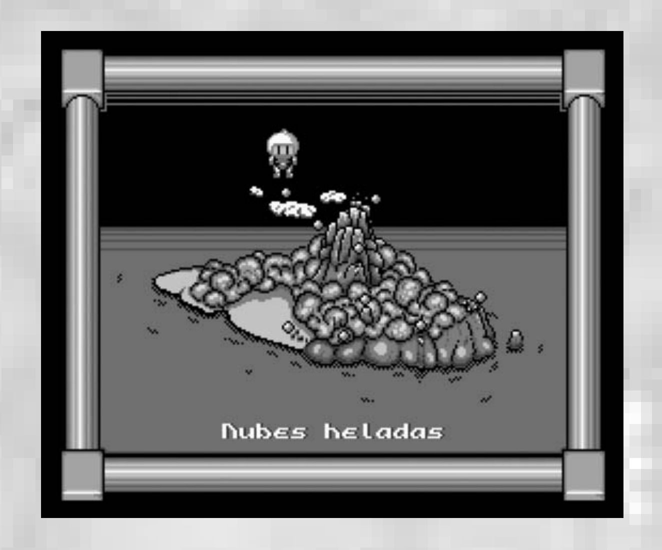
¿Quién no se ha echado unas partidas a cualquiera de las versiones de bomberman/dynablaster? Si bien en MSX las versiones "oficiales" no eran todo lo interesantes que merece un título indiscutible en el panorama tanto de consolas como de arcades, parece ser que esta versión de Team Bomba promete acabar con esos agravios. Un trabajo de cinco años bien merece ser considerado así, a priori. Y es que Bomberman (1983) y Bomberman Special (1986) sentaron las bases de lo que iban a ser los sucesores, pero se quedan lejos de los "modernos" a nivel de opciones. El primero presentaba un muñeco hecho por caracteres que ponía bombas (aquí llegó como Erik & The Floaters, al igual que al Spectrum) y el segundo ya aportaba al personaje carismático de la saga. No obstante ambos carecían de Battle Mode, posiblemente lo que ha hecho famoso a Bomberman por encima de todo.

REQUISITOS

Bombaman funcionará en cualquier MSX2 con al menos 128KB de memoria mapeada y 128KB de VRAM. Para los efectos sonoros tenemos varias posibilidades. Si sólo disponemos de PSG nos quedaremos sin la melodía, la cual puede sonar a través del MSX Music, del MSX Audio, de ambos o del MoonSound, siendo esta última opción la favorita por muchos. Así, podemos pasar de tener los efectos sonoros de siempre con el PSG a tenerlos sampleados con el MoonSound. ¿Nostalgia o modernidad? La decisión está en manos del usuario.

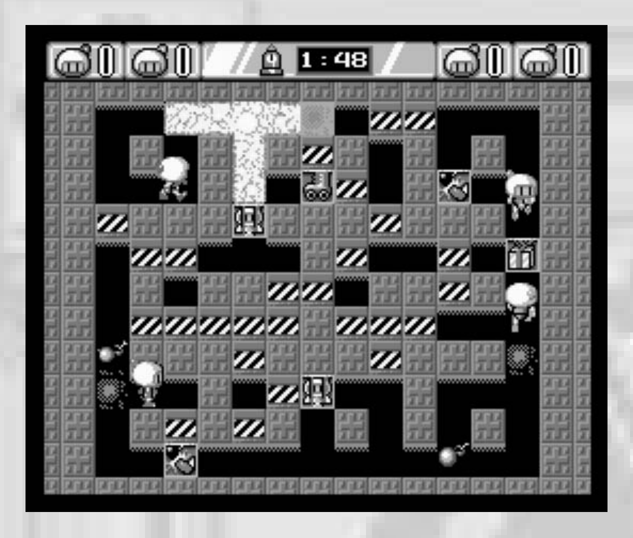
Una característica remarcable en Bombaman es la inclusión de la compatibilidad DOS1/DOS2. Como en otros juegos aparecidos con anterioridad (KPI Ball es un buen ejemplo) los chicos de Team Bomba han querido aprovechar la velocidad de los dispositivos de almacenamiento (unidades ZIPs, discos duros, memorias Compact Flash...) para dotar a Bombaman de una mayor agilidad de lectura de datos del medio físico a memoria. Así, copiando los ficheros de ambos discos a una carpeta y ejecutando BOMBAMAN.COM la velocidad de lectura anteriormente comentada aumentará. De todas formas, insertando el diskette y encendiendo el MSX el juego es autoarrancable.

Los poseedores de un turboR estarán de enhorabuena, ya que Bombaman hace uso del R800 para ejecutar código en muchas fases del juego, mejorando así el rendimiento del mismo.



CONTROLES

Bombaman está pensado para un máximo de 4 jugadores simultáneos. Cuando esto sucede, tenemos todas las posibilidades disponibles en uso. De este modo, dos jugadores harán uso de joysticks (de dos botones) mientras los otros dos se defenderán con el teclado. La disposición de teclas



para los usuarios de teclado será la siguiente:

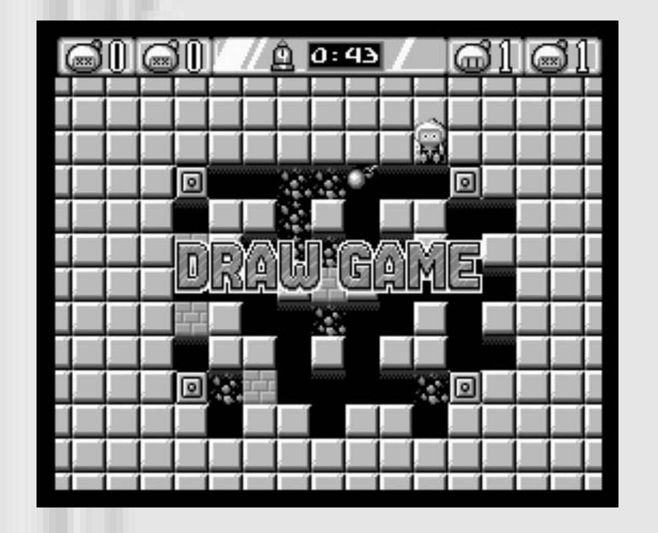
KB1:
Arriba – Cursor arriba
Abajo – Cursor abajo
Izquierda – Cursor izquierda
Derecha – Cursor derecha
A – Barra espaciadora
B – M

KB2:
Arriba – W
Abajo – S
Izquierda – A
Derecha – D
A – CTRL
B – SHIFT

En modo SINGLE, el jugador podrá optar entre usar KB1 o el joystick en el port 1 (JY1).
A continuación podéis encontrar qué uso podréis hacer de cada tecla durante la partida:

Tecla **Uso**
Cursor/Joystick Mover al jugador
Botón A Colocar una bomba

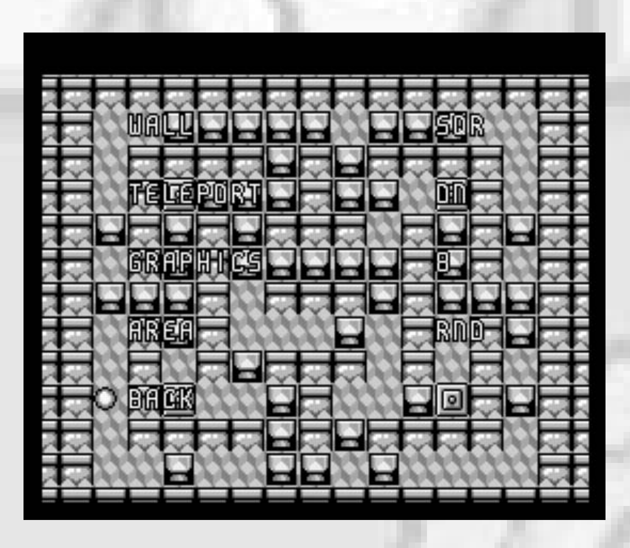
Hablar/pasar al siguiente párrafo
Botón B Detonar/lanzar una bomba (multiplayer)



Acelerar los textos
F1 Pausar
F2 Usar objeto especial
ESC Terminar la partida en curso

ITEMS

¿Qué sería de un buen bomberman sin sus correspondientes items? En Bombaman podremos encontrar un buen surtido de ellos. Ciertos items son exclusivos para un modo de juego u otro, aportando poderes o restándolos. Para el modo multi-player tendremos items específicos, orientados en su mayoría al espectáculo, ya sea para favorecerse uno mismo (o todo lo



contrario) o al resto de participantes. Para el modo single tendremos a nuestra disposición items más relacionados con el tiempo restante o con los poderes del personaje. Los items comunes a ambos modos hacen referencia al número de bombas que podremos poner en pantalla de una tacada (máximo dos) así como a su contundencia. Además modificarán habilidades del personaje tales como la velocidad de desplazamiento o su visibilidad.

MENU DE SELECCION

Una vez arrancado el juego podremos decidir entre unas posibles opciones o dejar pasar el tiempo para ver la intro. Intro que introducirá al jugador en la historia. A decir verdad es una intro como las de antes (screen 5, pequeñas animaciones)... y es que hace tiempo que no teníamos un juego entre manos de esta índole. Si no queremos esperar, podremos pulsar sobre la opción INTRO.

Lo primero que deberíamos seleccionar es, sin duda, SETTINGS. Tendremos a nuestra disposición el cambio de idioma (los hispanohablantes podemos estar de enhorabuena de que se hayan acordado de nosotros), de cartuchos de sonido a usar y algo muy a tener en cuenta, el SCREEN POSITION (también conocido como SET ADJUST). Y es que no todos las TV se ajustan horizontalmente bien a los 60 Hz, así que es todo un detalle acordarse.

A nuestra disposición tendremos un modo CUSTOM donde podremos cargar y jugar a los niveles anteriormente creados con el editor de pantallas (soft PC para windows contenido en el mismo CDROM que acompaña al juego) o directamente descargados desde la página web del Team Bomba: http://www.bombaman.net

Para finalizar, tendremos los modos clásicos de juego: NEW GAME, para empezar una nueva partida en modo historia, PASSWORD para continuar una partida y MULTI PLAYER para competir en modo batalla.

NEW GAME

En un primer momento, se pondrán a nuestra disposición dos niveles de dificultad, EASY y NORMAL. Tras finalizar con éxito el modo NORMAL, hará aparición un nuevo desafío, el nivel HARD. Entre ellos, además del nivel de dificultad, podremos encontrarnos con ciertas diferencias, así que jugarlos todos es un deber. No sería del todo positivo destriparos el juego, por lo tanto, descubrir qué cambia en cada uno de ellos deberá ser propiciado por la pericia del jugador.

El famoso modo historia, toma aquí una mayor relevancia. Y es que en Bombaman hay un guión a seguir, incluso pueblos, más parecidos a los que podríamos encontrar en cualquier RPG. Fases de transición entre pantallas, donde encontrar información, avanzar por el mapeado, etc. Este hecho puede considerarse un arma de doble filo... por una parte, quien quiera únicamente echarse unas partidas bien seguro que le parecerá una pérdida de tiempo tener que empezar en el pueblo y hacer el recorrido hasta encontrar la salida. Y estos casos existen, por lo que se podría haber pensado más en ellos. ¿Quizás un modo en el que las pantallas se fuesen sucediendo sin más? De todas formas, siempre se puede hacer uso de los PASSWORDs para tal fin.

Bombaman consta de 6 mundos, cada uno con 11 fases, siendo la undécima la fase del enemigo final (final boss). Sólo recalcar que encontraremos tres tipos de fases. En las primeras, pasaremos a la siguiente tras matar a todos los enemigos de la misma. En las segundas, deberemos recoger todas las gemas o destruir mediante bombas los cristales. El tercer tipo son las ya mencionadas donde un enemigo final nos esperará. Tras acabar con él, pasaremos al siguiente mundo. Cabe mencionar que el número inicial de vidas será de cinco.

PASSWORD

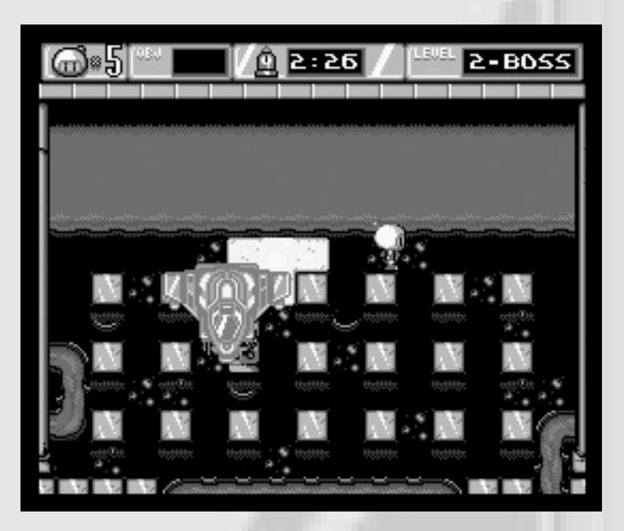
No hay mucho remarcable en este punto, con la salvedad de explicar las diferencias que podremos encontrarnos al empezar en una u otra fase de un mundo y si empezaremos (o no) con objeto/s especial/es. Además, estos password serán distintos dependiendo del nivel de dificultad escogido.

Así, un password aparece cada vez que acabamos la fase 5, la 10 o la 11 (enemigo final) de un mundo. Pero hay que tener claro que, si continuamos en la fase 1 de un mundo, conservaremos 5 vidas, mientras que si lo hacemos desde la fase 6, sólo tendremos 3. También se nos dará un password distinto si manteníamos el objeto especial al acabar la fase (pueden ser dos objetos en nivel Hard). Aunque todo esto pueda parecer lioso, tras unas cuantas partidas se comprende perfectamente.

MULTI PLAYER

Lo que distingue a Bombaman por encima de la mayoría juegos, es sin duda la posibilidad de que varios usuarios se embarquen en un combate todos contra todos. En MSX estos juegos la verdad es que no abundan, pese a haberse construido hardware a tal efecto, como el Ninjatap... ¿quién no recuerda el Magical Labyrinth?

Pues bien, el modo multiplayer de bombaman, no va a defraudar a sus incondicionales. Diversión, piques y emoción a partes iguales donde tienen cabida hasta 4 jugadores. Si por algún motivo hubiera una o más vacantes, éstas pueden ser



reemplazadas por la máquina. Huelga decir que el sistema de juego es bastante sencillo y ganará la partida el que sobreviva a la batalla (el combate será considerado nulo si hay más de un ganador cuando el tiempo llegue a 0 si hay límite para el mismo).

Como la opción del límite de tiempo, podremos configurar otras antes de empezar un combate. Un ejemplo es el número de *rounds* que deberá ganar un jugador para ser considerado el vencedor de la partida. También decidiremos sobre el escenario o la música que se utilizarán y, como novedad, podremos decidir si se producirá un *flash* en la pantalla al estallar cada una de las bombas. Todo eso unido a los items que podremos encontrar, aseguran horas de vicio contra otros usuarios.

ASPECTOS TÉCNICOS

Antes de abordar tan espinoso tema, hay algo que debería ser revisado anteriormente. Mucha gente podrá pensar que

Bombaman es un juego tremendamente complicado o de un nivel de dificultad elevado. El que escribe también lo ha visto así en un principio, pero pronto ha descubierto que tal nivel es el ajustado, siempre y cuando, por lo menos al principio, el uso de passwords sea el menor posible. Es un juego con el que la habilidad del jugador se incrementa a cada partida. Saber cómo derrotar a cualquiera de los enemigos que aparecen o cómo acabar una pantalla con éxito será fruto de jugar bastante. Con el avanzar de las pantallas (y con ello los mundos), el aprendizaje será demostrado al poco tiempo. Así que, la mejor forma de poder pasar las pantallas más difíciles, será haber pasado por las anteriores cuantas veces sean necesarias.



Todo esto no es sólo obra del programador, Arjan Bakker. Para conseguirlo, ha tenido la inestimable colaboración de la cara española en Team Bomba, Jesús Pérez Rosales. Casi un desconocido para la mayoría, apareció para demostrar que tiene traza para dotar a los escenarios de su sello característico. De hecho, ya hace tiempo creó para otros juegos (como King's Valley) escenarios para el disfrute del personal. Puede decirse que es un auténtico especialista tras los años.

La jugabilidad es uno de los puntos más importantes a considerar al encontrarnos ante un soft novedoso. Ésta aglutina la suavidad, la ajustada dificultad (que ya ha sido comentada) y la respuesta al usuario. Bombaman está programado en ASM, pero haciendo uso de Screen 5, lo que puede hacer mella en este punto crucial. Además, la no utilización de sprites lo dificulta más si cabe. Pese a no tener una suavidad tan espectacular como otros juegos (por las razones obvias descritas) no desmerece en absoluto. Es más, hay un equilibrio importante durante la partida, donde las ralentizaciones no son apreciables (por lo menos en el modo historia). Está claro que no alcanzará las cotas de suavidad de títulos como Dream On (Screen 4 y sprites), pero por el contrario, los personajes ganan en vistosidad.

Y es aquí, en la vistosidad, donde Bombaman más ha cambiado en los últimos meses antes de hacer su aparición en el mercado. La magistral mano de Sutchan ha hecho posible una mejora sustancial de los gráficos, a nivel de gráficos y paleta. Aquí demuestra una vez más que es el grafista más en forma del panorama MSXero, por lo menos a nivel europeo, logrando "revivir" la apariencia para unos un poco apagada de la versión existente hasta entonces. Así, ha sido el complemento ideal de última hora del grafista original, Robert Vroemisse. Decorados que cumplen su acometido y enemigos vistosos a la par que odiosos (cuando juguéis me daréis la razón).

El apartado sonoro de Bombaman merece una atención especial. Cabe decir que no dispongo de un MoonSound por el momento. Me he tenido que "limitar" a escuchar la OST con FM+MM+PSG. La calidad compositiva no desmerece en absoluto pues las canciones son las mismas, pero he tenido la oportunidad de escuchar las mismas en el MS y ahora me veo en la necesidad de comprar uno (Jorrith Schaap es el culpable de que tenga que gastar dinero...).

Tras haber escuchado las melodías con los distintos cartuchos, he llegado a mis propias conclusiones. La primera es, que mi Music Module no está en las mejores condiciones y necesita una revisión. Suena muy bajo en el turboR además de emitir ciertos pitidos que indican seguramente que la ampliación de 256KB no está como cuando fue implantada. Bromas aparte, quiero creer y en esto coincido con otros usuarios, que las melodías fueron primero creadas para MS y luego convertidas a FM+MM. Hay que hacer hincapié en esto porque normalmente no sucede así. Se tiene la impresión que el MS es siempre "demasiado opcional" lo que puede hacer que las melodías no estén "diseñadas" a conciencia para el mismo y su calidad sea inferior a lo debido.

De este modo, la banda sonora en MoonSound (para el que no lo tenga podrá escucharla desde el CD adjunto) es, bajo mi punto de vista, la mejor que se ha hecho para nuestro sistema y para ese cartucho en concreto. Además, los efectos sonoros sampleados crean una nueva atmósfera, no antes vivida en el software del estándar. Nos acercan a otros sistemas más modernos en el tiempo.

Pues bien, atendiendo a todo lo esgrimido anteriormente y a modo de conclusión, destacar que estamos ante un muy buen juego que desprende trabajo por todas partes. Pocos puntos flacos vamos a encontrar, aunque no niego que quizá si se profundizara más, podría hallar algunos más. Y es que han podido quedar muchas cosas en el tintero, tanto para bien como para mal.



Lo que sí es seguro que se trata de un juego que no dejará indiferente a nadie. Eso sí, todavía no he recibido mi original, así que no puedo hacer crítica de la presentación. Pero sabiendo que la OST se halla en MP3, hubiera preferido algún otro formato reproducible en MSX, como grabarla en pistas de audio. Pero bueno, a caballo regalado...

ENTREVISTA AL TEAM BOMBA

La primera pregunta puede resultar obvia, pero me parece siempre obligada. ¿Por qué otro Bomberman? ¿Qué opináis de las anteriores versiones "amateurs" que del clásico se habían realizado para el sistema?

Jorrith: En un principio, Bombaman fue pensado como un sencillo juego multijugador que sería añadido en la FutureDisk. En FutureDisk aparecían pequeños juegos basados en la temática de la revista. Bombaman iba a ser uno de ellos, donde podías hacer volar a los contrincantes al estilo South Park.

Esa FutureDisk se pospuso y mientras tanto Bombaman se hacía más y más grande. Éste fue el motivo por el que decidimos transformarlo en un juego completo. A tu segunda pregunta, sólo he visto los Bomberman de Paragon y, francamente, no me gustaron mucho. ¡Eran más juegos tipo puzzle que el juego de acción que todo Bombermán debería ser! Por supuesto, nada supera a Bombaman. ;)

Arjan: Del mismo modo que Jorrith, las únicas versiones que conozco para MSX son las de Paragon. Tampoco me gustaron mucho, ni su modo de juego ni técnicamente. La velocidad del juego variaba de vez en cuando lo que no era bueno.

JPR: No había jugado a ningún Bomberman, salvo el Eric & The Floaters, cuando empecé a colaborar con el Team Bomba. Ese juego está bien pero, como los antiguos, era también un poco repetitivo. Recuerdo que Jorrith me sugirió probar algunos Bomberman para que me imaginara lo que Bombaman podría llegar a ser; es divertido recordar ahora que no me apetecía probar Bombermans porque pensaba que podría copiar directamente los mapas a Bombaman; de todas formas,

al final probé una versión moderna y me desanimó un poco porque pensé que los Bomberman, precedidos de su fama, serían muy adictivos. En cambio, esa versión ofreció muy buenos graficos pero un bajo nivel en lo que a Inteligencia Artificial se refiere. Además, los mapas eran demasiado

sencillos. No puedo decir que nuestro juego sea genial, una excepción (soy una parte subjetiva), pero ahí queda mi experiencia como jugador.

Robert: Hay una versión bastante buena para GFX9000, pero no tiene modo para un jugador. Además, hay algunos clones con muchos puzzles, pero no me gustan. Eché en falta la acción en aquellas versiones.

El Team Bomba lo formáis cuatro personas. ¿Podríais contarnos algo de cada uno de vosotros? ¿Cómo se forjó la unión? ¿A quién corresponde la idea original del proyecto?

Jorrith: Yo soy sobre todo el compositor del Bombaman, habiendo hecho la mayoría de la banda sonora y de los efectos de sonido. También hice algunos gráficos (fuentes sobre todo) y me nombré además director del proyecto; los otros componentes del grupo me hicieron además supervisor de la labor de traducción. Me uní muy pronto al proyecto: siendo Arjan miembro de la plantilla de FutureDisk y yo compositor de éste, era obvio a dónde se debía ir para la música del juego. La idea del proyecto original fue surgiendo, más o menos, supongo; culpad a Koen Dols :)

Arjan: Qué va :) Yo fui el que vino con el concepto original de la panda de South Park batallando al estilo Bomberman (puesto que se suponía que yo haría una edición de FutureDisk ambientada en South Park). De cualquier modo, lo programé todo, lo que significa que hice el código del juego, del editor y de todas las herramientas que nosotros solemos usar. También creé algunas canciones mejoradas por Jorrith.

JPR: Cuando me uní al proyecto el juego estaba ideado y altamente desarrollado. Conocí a Arjan y a Jorrith chateando por diversión en el canal del #msx del IRC; gente como Francisco Álvarez y otros usuarios del MSX hicieron que me dejase ver frecuentemente por el canal y Jorrith me animó a ayudar al Team Bomba a hacer niveles para Bombaman. Hacer un videojuego había sido un sueño

imposible para mí pero esta gente me demostró que uno puede cooperar en cualquier cosa siempre que cada uno haga su propio trabajo y ayude al resto del mejor modo posible. Me encanta editar niveles para videojuegos por lo que me uní e hice uso del motor ya creado (la versión beta del editor) para hacer fases. ¡Y al final lo logramos!

Robert: Se me pidió que dibujase algunos gráficos simples para un modesto juego allá por 1999. Eso evolucionó a Bombaman. Las ideas de incluir jefes finales y una línea argumental fueron mías. A todo el mundo le gustó y lo demás queda para la historia.

En España hace mucho tiempo que se discute si es mejor un juego que aproveche al máximo los recursos propios del MSX

antes que los "extras". En vuestro trabajo se ve claramente que optáis por lo segundo. ¿Cuáles son las principales razones para vuestra elección?

Jorrith: Más o menos te hartas con un sistema MSX2 básico, y ya que hay mucha gente con dispositivos adicionales como un Moonsound, ¡sería una vergüenza no usarlo! Por lo tanto, básicamente, ¡la razón principal fue hacerlo interesante para nosotros mismos! LLegamos incluso a considerar el aceptar exclusivamente Moonsound y no MSX-Music ni MSX-Audio, pero al final decidimos añadirlo.



Arjan: Lo único que estamos utilizando como 'extra' es el Moonsound (a menos que quieras llamarle 'extra' también al MSX-Music o al MSX-Audio, pero ya se consideran parte del estándar). A mí no me importaba que en realidad aceptásemos exclusivamente Moonsound en Bombaman, pero Jorrith decidió del otro modo :) De cualquier modo, la música y los efectos de sonido con Moonsound te transportan absolutamente a otra dimensión de juego, ¡por lo que es mucho mejor así! ¿No nos crees? ¡Puntéa las pistas del CD! Por cierto, también aceptamos otros EXTRAS como DOS2, HD, CompactFlash y Moonsound con menos de 128kb de SRAM.

Robert:oonsound y DOS2 se aceptan como extras. Puedes jugar en un MSX2 normal sin ningún sonido extra, pero los extras le ponen al juego la guinda al pastel.

Desde el punto de vista de la programación... ¿ha habido algo realmente complicado que quisierais destacar?

Arjan: Hacer que todo funcionase fue el mayor problema... Puesto que no tengo DOS2, me era difícil hacer que el juego fuese compatible con DOS2, por lo que le pedí prestado a Jorrith su Turbo R para poder probar cosas en él. Afortunadamente pude arreglarlo, pero eso me llevó a otro fallo en el formato de compresión que nosotros usábamos (POPCOM). De ahí que decidiese fabricar

nuestro propio formato, BitBuster, que está libremente disponible en nuestra página web. Es difícil de manejar todo lo que puede ocurrir en el juego (especialmente en el modo Multiplayer) ya que todo lo que sucede lleva a otro suceso. Eso es lo que consigues si no diseñas completamente el juego antes de programarlo :P. No fue fácil tampoco crear una IA decente para el modo Multijugador. Lo rematé con algo que es semi-decente, es bueno tener jugadores con IA si no puedes jugar con cuatro jugadores humanos, pero se pueden comprobar claramente las limitaciones si tienes cuatro jugadores controlados por el ordenador.

El apartado gráfico en este tipo de juegos necesita claridad para poder distinguir cada zona del escenario y evitar confusiones (objetos, personajes, bloques destruibles...). ¿Ha llevado mucho tiempo lograr este objetivo?

Robert: Sí. El objetivo tenía que ser el mismo en cada mundo, por lo que usé colores brillantes estándar para ello. El de los gráficos, como los gráficos de los mundos y demás se hicieron con una apariencia más mate para que los diferentes elementos fuesen claramente distinguidos. Para ello se precisaron mucho los colores.

Musicalmente hablando, Bombaman ha tenido críticas muy positivas desde el comienzo (sobretodo la versión moonsound). Si es cierto que las versiones se hacían para moonsound y luego para FM (y no al revés)... ¿ha costado mucho acostumbrarse al poder del moonsound para luego volver atrás?

Jorrith: Pues sí, es verdad. Una razón muy buena para esto es que es más fácil hacer una canción guapa desde el Moonsound empezando desde cero y convertirla luego a FM sin demasiada pérdida de calidad que de la manera inversa. Nos pareció realmente bueno hacer uso de todas las ventajas que el Moonsound podía ofrecer, la música en FM era simplemente un buen extra para la gente sin Moonsound. Por lo tanto, ¡es mucho mejor el material para el Moonsound! Usé una pequeña herramienta para convertir los ficheros del Moonsound a MBM, haciendo uso de mucho tiempo para preprocesar el fichero MWM y luego postprocesar el fichero MBM para rellenar las cosas que el conversor no rellenaba. No es muy divertido hacerlo con unas 60 canciones, ¡pero es mucho mejor que reescribir completamente todo!

La jugabilidad es uno de los factores más decisivos cuando se valora un juego. ¿Qué podéis decir de ella en Bombaman?

JPR: El Bomberman clásico fue el factor principal puesto que el motor de la jugabilidad se basaba en aquél. No obstante, algunas sesiones de devanarnos los sesos que llevó a cabo Team Bomba cada vez que nos encontrábamos en el IRC pateó un buen puñado de conceptos de Bomberman y surgió el material propio. Podríamos decir que el origen está tan lejos del juego actual que ahora no podríamos considerarlo un Bomberman clásico; el modo historia tiene 3 objetivos diferentes, más de diez tipos diferentes de IA, mapas originales, passwords, el objeto especial... La atmósfera de RPG es otra fresca y nueva característica (esta parte pertenece a Robert en su totalidad): da una variedad inusual. Adema´s, el editor proporciona horas de juego creativo, el modo Multiplayer está muy cuidado (un montón de opciones nuevas)... Así pues, aunque el género ya estaba creado, los elementos de jugabilidad incluidos son originales. ¡Lo mismo va para los gráficos y para la banda sonora, por supuesto!

Subrayando los principales factores de jugabilidad en Bombaman, yo señalaría los nuevos niveles, el sistema de passwords, la excepcional IA, los jefes, incluso la canción que acompaña cada nivel no está ahí de casualidad. El 'aprendizaje progresivo' que comentabas en el artículo es una de las cosas de las que estoy más orgulloso, tanto como la longitud del juego (que incluye niveles ocultos y algunas sorpresas). Bombaman no supone ver una película, ¡pero puedes soñar con él!

Hasta hoy, ha aparecido alguna actualización para mejorar el juego original (nuevos idiomas, bugs, ...). ¿Creéis que os encontraréis con algún nuevo problema en los próximos meses?

Jorrith: Esperemos que esta sea la versión final de verdad. De hecho, no creo que haya más problemas gracias a que mucha gente ha comprado el juego y nos ha proporcionado información acerca de fallos y cosas que nunca hubiésemos probado nosotros mismos. Con todas estas pruebas y partidas fuera del guión la mayoría del tema debería de quedar listo :)

Una vez acabado este largo proyecto... ¿os queda energía para algo más? ¿Seguirá el Team Bomba unido para una nueva producción? ¿Podéis comentar un poco el futuro?

Jorrith: Team Bomba continuará en el futuro, deseamos volver a la escena del MSX pronto con otro juego. Será un juego para MSX2, Graphics9000 y Moonsound, y ésa es toda la información que podemos proporcionar por el momento. Podría valer de algo visitar en poco tiempo nuestra página web http://www.teambomba.net para más información.

Robert: Ahora mismo estamos planeando un juego para Gfx9000, aunque no creo que pueda contribuir demasiado ya que pronto seré padre; no obstante, cuando tenga tiempo, volveré a mover el culo.

Para finalizar la entrevista, querría pediros qué opinión tenéis sobre la situación del sistema en Holanda, a día de hoy.

Arjan: La situación se está poniendo cada vez mejor. Recientemente, un pequeño grupo de gente del MSX ha vuelto de nuevo al mundo del MSX, lo que ha significado nuevo material como DISK #8 (magazine holandés). También hay algunos juegos en desarrollo, ¡uno de ellos parece que promete de veras!

Jorrith: Mola mucho ver que el 'revival' está ocurriendo de veras, con más gente perteneciente a la antigua generación del MSX volviendo y contribuyendo para con el MSX. El MSX Resource Center está desempeñando una función importante al respecto, no sólo en Holanda sino también a nivel internacional. Esperemos que este creciente interés continúe, para que al menos tengamos gente a la que venderles nuestro nuevo juego ;)

Francisco Álvarez



De izquierda a derecha: Robert Vroemisse, Jesús Pérez Rosales, Jorrith Schaap, Arjan Bakker

Ancient Ys Vanished Omen

(La maldición del antiguo y desaparecido pueblo de Ys)
1ª parte

Guía por: Zanac2



El poblado de Minea

Minea siempre ha sido un lugar bastante tranquilo y apacible. Las murallas construidas alrededor de este poblado han protegido a sus habitantes de amenazas externas; pero parece ser que algo no anda bien ya que últimamente la gente ha notado la presencia de una gran multitud de seres extraños fuera de las murallas. Todo parece indicar que la tierra de *Ys* está en peligro y que pronto se brindará una guerra contra el mal.

Nuestro joven guerrero *Adol Kristy* comienza su aventura en el pueblo de *Minea*. Acabamos de llegar de viaje y no conocemos el lugar, así que nos dirigimos a la taberna para tomar un trago y charlar un poco con la gente de por aquí. Al principio no somos bien recibidos, ya que últimamente merodean muchos ladrones por la zona, pero en seguida nos ganamos la confianza de los presentes. Un musculoso guerrero llamado *Kyalek* nos invita a un trago y nos pregunta si sabemos algo de un árbol gigante conocido con el nombre de *Roda*, que se encuentra a las afueras, cerca de aquí. *Donis*, el tuerto, también nos invita a beber y nos dice que le encantaría conseguir un anillo de zafiro que vio en la tienda de objetos usados.

Ya hemos bebido demasiado por lo que salimos de aquí y damos una vuelta por el pueblo. La gente está preocupada y habla de muchas cosas. Cuentan que hace poco apareció un misterioso hombre encapuchado al que lo rodeaba un aura de color verde y que los niños vieron monstruos tras las murallas. También se cuenta la leyenda de una espada legendaria en algún lugar no muy lejos de aquí. Si hablamos con toda la gente del pueblo seguro que conoceremos a una chica que está preocupada porque su padre está enfermo. Él es el alcalde de la aldea de *Zeppic*, situada hacia el norte, a poco camino de aquí. En la torre noreste de la muralla también se halla otra chica: es la poeta del lugar. Está disgustada porque unos ladrones le han robado su armónica. Ella no puede inspirarse sin su querido tesoro.

La pitonisa del pueblo espera a un héroe que salvará a la tierra de *Ys* de su fatal destino. ¿Será nuestro protagonista el guerrero al que esperan? No podemos perder el tiempo, así que nos dirigimos hacia la armería. Como no disponemos de mucho dinero sólo podemos hacernos con la **espada corta**, con la que podemos abatir a pequeños *Goblins* y *Orcos*. En la tienda de al lado compramos también una **cota de malla** para rebajar un poco los daños sufridos en los combates.

Nuestra primera batalla

Una vez equipados con todo lo necesario para afrontar los combates contra el enemigo, abandonamos el pueblo. Tomamos rumbo hacia el norte cruzando las llanuras de *Minea* donde lucharemos contra diferentes seres. Un frondoso bosque nos impide avanzar en esta direc-


Pitonisa


Armero


Armero


Taberna


Enfermera

ción, así que cruzamos el puente que hay situado al este. Al otro lado del río hallamos un lago donde paramos a tomar un trago; examinamos bien la orilla y encontramos un **amuleto de plata**.

Contentos con nuestro valioso hallazgo caminamos hacia el este y encontramos un acceso entre los árboles que nos lleva directo a la aldea de *Zeppic*. Allí visitamos la casa donde vive el alcalde; es el anciano enfermo del quién nos habló la chica de *Minea*. Él nos pide, por favor, que le ayudemos a encontrar sus campanas de plata que le fueron robadas y que ahora se encuentran en algún lugar de *Ys*. La gente comenta que desde que cerraron la mina de plata, se han robado muchos objetos de este precioso metal.

Volvemos tras nuestros pasos, camino de *Minea*, mientras conseguimos el suficiente dinero para comprar un **escudo pequeño** que cuesta 700 monedas en la tienda del armero; también podemos optar por vender el amuleto de plata que hemos encontrado, en el pueblo nos darán 2000 monedas por él, pudiendo escoger entonces el **escudo mediano** si queremos que todo sea un poco más fácil. Otra posibilidad para obtener más dinero es la de comprar el anillo de zafiro en la tienda de objetos usados y vendérselo a *Donis*, el hombre tuerto que se encuentra en el bar; éste nos lo comprará por 1500 monedas, así que ganaremos 500 monedas, un buen negocio. Lástima que solamente podamos hacerlo una vez.

Una vez equipados con nuestro escudo, hacemos una visita a la pitonisa. Ésta nos habla acerca de los libros de *Ys,* unos libros que narran la historia de estas tierras, escritos después de la destrucción. La bola de cristal le revela que uno de los libros se encuentra en el templo de la montaña. La pitonisa pide por favor que lo recuperemos. Prometemos encontrarlo y por nuestra valiente decisión la mujer nos ofrece un **cristal** y nos asegura que nos guiará en nuestro camino. Nuestra experiencia aumenta en 100 puntos.

El primer libro de Ys - El templo de la montaña

El templo que guarda el primer libro se encuentra en la montaña que se alza frente a la aldea de *Zeppic*, así pues marchamos hacia allí. En una de las casas de la aldea se encuentra la vieja *Jehva;* se sorprende al ver que tenemos el cristal de *Sara* (el que nos regaló la pitonisa) y deduce que estamos buscando los libros de *Ys*, por lo que nos proporciona una llave. Esta llave nos servirá más adelante para abrir una puerta que hay en el interior del templo de la montaña, pero antes de dirigirnos hacia allí necesitamos ganar algo más de experiencia y dinero para comprar una espada mejor, por lo que lucharemos hasta reunir las 2000 monedas que cuesta la **espada larga** (que nos servirá para derrotar al gran enemigo que se esconde en el interior del templo). Una vez obtenemos esta arma, volvemos a *Zeppic*.

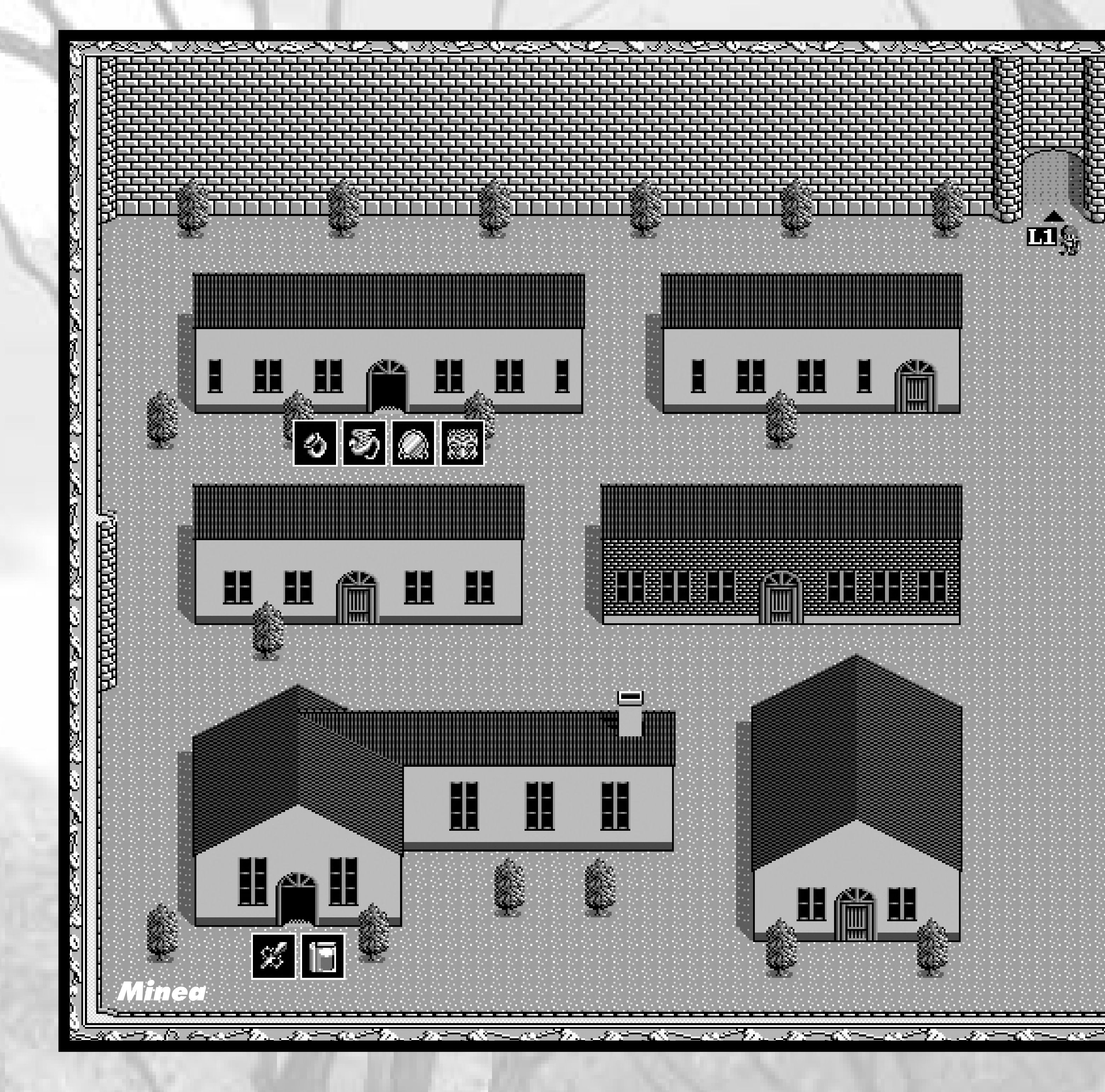
En la parte norte de la aldea existe un camino que sube por la ladera de la peligrosa montaña y que nos conduce directamente hasta el templo. Al entrar en el interior del edificio observamos una pequeña sala con varias estatuas. Nos extrañamos al ver que no existe ningún otro acceso o puerta hacia otra sala y nos llama la atención que una de las estatuas es diferente al resto. Nos situamos frente a la estatua y automáticamente somos transportados al interior del templo. Hay diversas salas pero por suerte parece que no hay ningún ser hostil. Buscamos por todos los rincones y descubrimos dos baúles. Uno de los baúles permanece cerrado mientras que el otro contiene un **rubí**. Examinamos mejor el lugar y hallamos una puerta de cristal que podremos abrir con la llave que nos proporcionó la vieja *Jehva*. Hay que vigilar la experiencia que hemos conseguido antes de continuar adelante, un nivel de experiencia cercano a 1000 es recomendable.


Anciano de Zeppic


Jehva


Anticuario

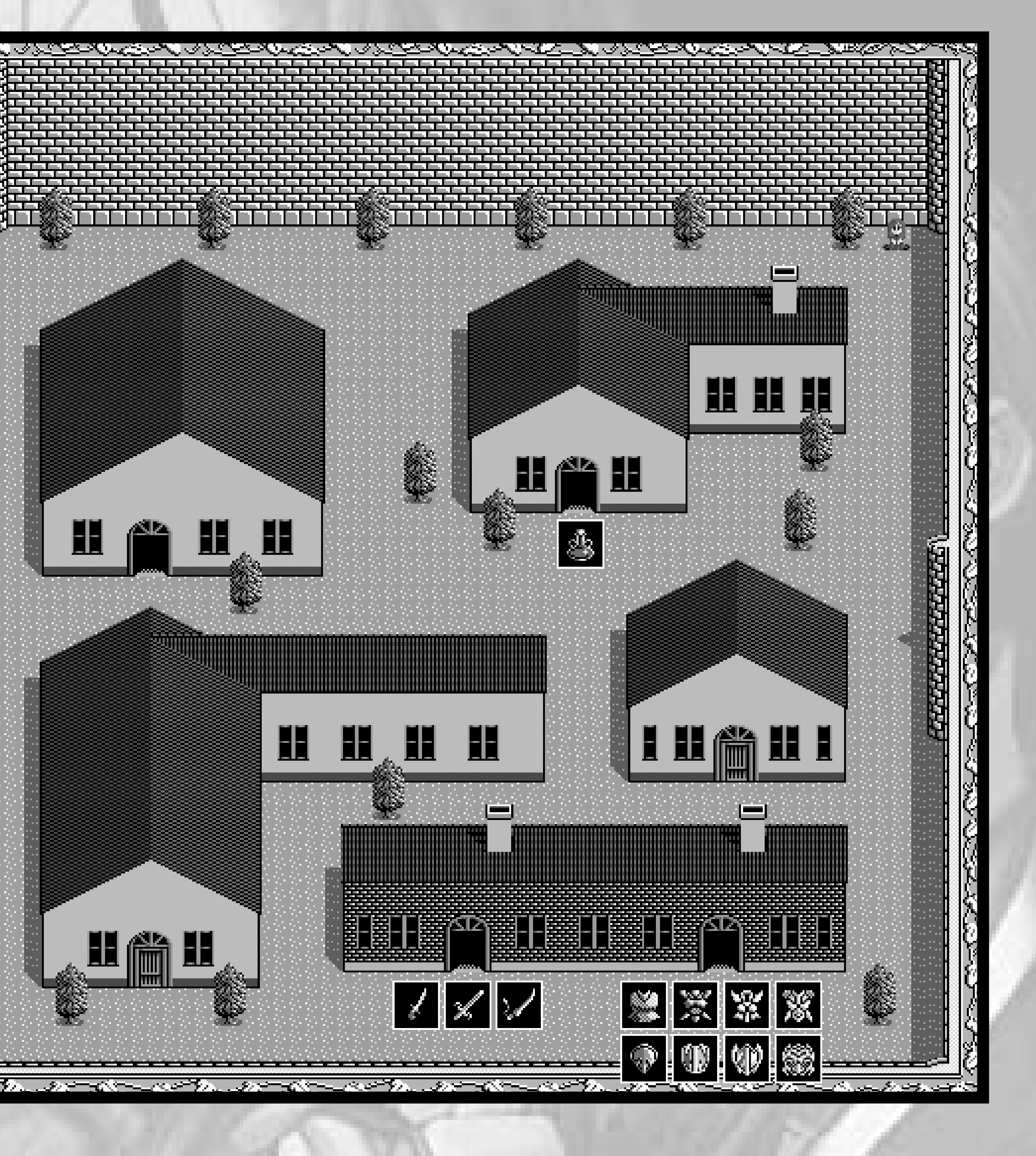

Jehva y Feena



Una vez decididos, abrimos la puerta de cristal y accedemos a una sala adornada por pequeñas estatuas y por un enorme grabado en la pared. Examinamos bien el grabado y de repente las pequeñas estatuas empiezan a escupir fuego. Ante nosotros aparece un hechicero que tiene la habilidad de aparecer y desaparecer; para derrotarle esperamos a que sea visible y arremetemos sobre él con todas nuestras fuerzas. Después de varios impactos conseguiremos derrotarlo; justo en ese momento se abrirá ante nuestros ojos la pared de enfrente, pudiendo así continuar por un nuevo acceso secreto hacia lo más profundo del templo.

Bajamos por unas escaleras que nos llevan a un nivel inferior. Mantendremos nuestra espada firme porque aquí se esconden nuevos enemigos. Nos dirigimos hacia el este y hallaremos dos celdas, pero no podremos abrirlas. Vamos hacia el sureste y allí descubriremos un cofre que contiene un **collar**. Seguidamente caminamos un poco hacia el oeste y nuevamente encontramos otro cofre. Éste contiene la **máscara de ojos**, un objeto mágico que nos será muy útil un poco más adelante. Regresamos a la primera zona, donde se encuentra la entrada a este nivel y desde allí nos dirigimos al suroeste hasta llegar a unas escaleras que nos conducen más abajo todavía, a un segundo piso bajo tierra.

Caminamos un poco por el pasillo y entramos por la primera puerta que vemos; seguimos adelante y pocos pasos después hallamos un cofre que contiene la **llave de la celda**. Regresamos al primer piso y abrimos la celda de la derecha; allí se encuentra *Feena*. Nos da las gracias por haberla liberado y nos dice que no nos preocupemos en acompañarla porque



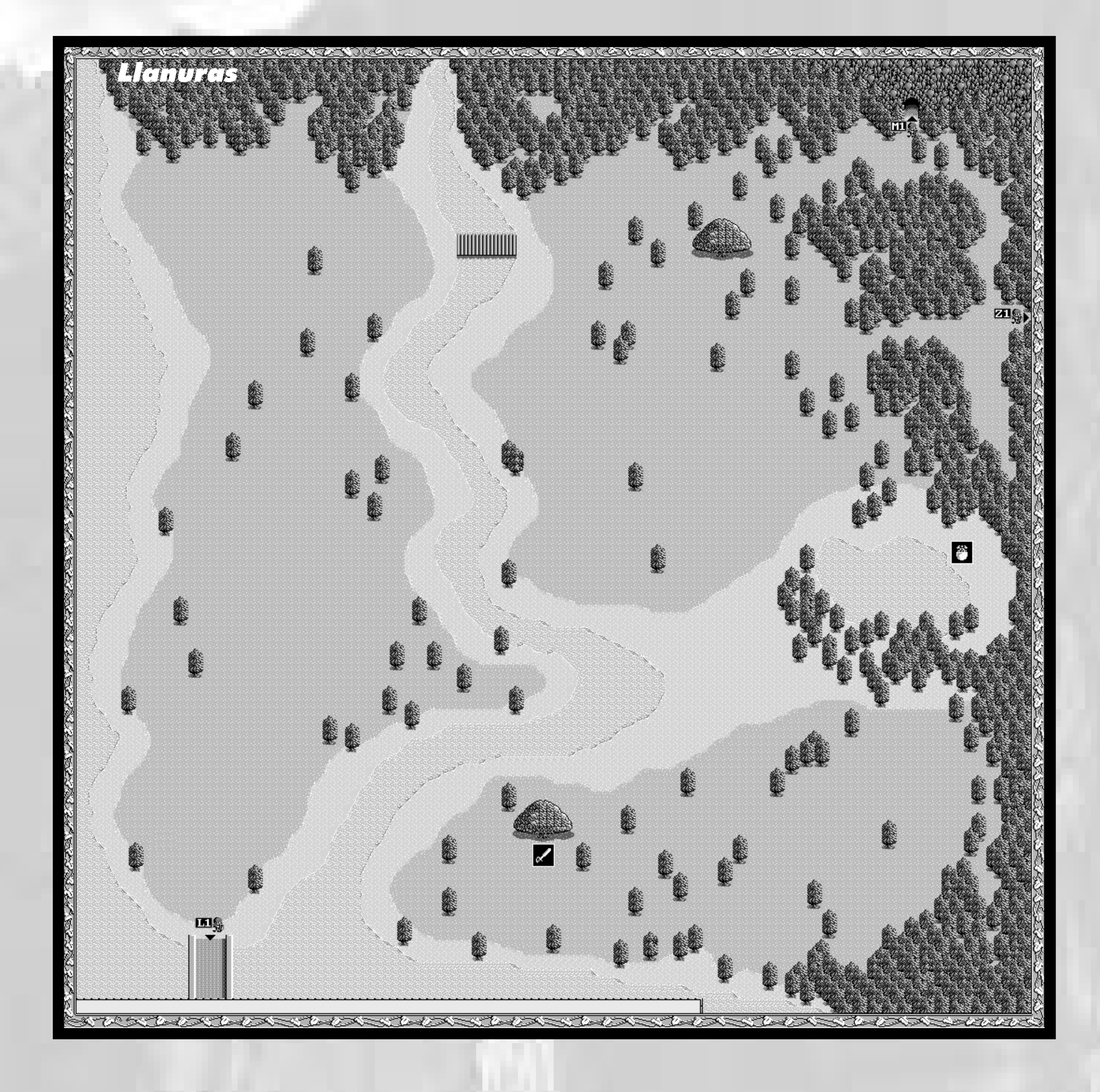
ella sabe cómo huir de allí sin ser vista. Ganaremos 500 puntos de experiencia por nuestra hazaña.

Nuevamente vamos al piso inferior, caminamos un poco y en esta ocasión entramos por la segunda puerta. Investigamos un poco ya que por aquí podremos encontrar dos cofres; uno con **la llave del tesoro** y otro con las **campanas de plata** que buscaba el viejo alcalde de *Zeppic*. Optamos por volver atrás, prácticamente a la entrada del templo, donde está el cofre que no podíamos abrir y que ahora, con la llave que acabamos de obtener, podemos ver el contenido del mismo, obteniendo el **anillo mail** que nos ayudará en los combates. Para aprovechar el camino nos acercamos a *Zeppic* y entregamos las campanas de plata al alcade, el cual nos ofrecerá otro anillo, el **anillo del poder**.

Regresamos al interior del templo y nos dirigimos al punto donde hallamos la llave del tesoro para luego dirigirnos al sureste donde bajamos otras escaleras.

Llegamos a un nuevo nivel, donde debemos encontrar dos llaves. Caminamos hacia el extremo situado más al noroeste y encontramos un nuevo baúl con la **llave de marfil.** Seguidamente nos dirigimos al extremo suroeste donde nos encontraremos con una nueva estatua teletransportadora, sin embargo, no la cruzamos y seguimos el pasillo en dirección este hasta llegar a una segunda. Esta estatua nos llevará a otra zona y allí podremos encontrar el **escudo de plata**. Volvemos hasta la primera estatua teletransportadora que vimos y cruzamos a través de ella.

Aparecemos en otra zona del templo,



aquí podemos encontrar una poción sanadora en un baúl que hay hacia el oeste. Utilizamos de nuevo la máscara de ojos y una nueva puerta secreta al lado de la estatua azul aparece ante nuestros ojos. Aquí descubrimos otras dos estatuas tele-transportadoras, cruzamos por la que se encuentra más al este y accedemos a un lugar secreto donde se halla la **llave de mármol**.

Regresamos de nuevo a través de la estatua y cruzamos por la otra, la que se encuentra cerca de la puerta secreta. Un largo pasillo de columnas nos separa del gran enemigo de este templo. Armados de valor utilizamos las dos llaves encontradas para abrir las dos puertas que encierran la bestia. Ahora nos encontramos en una estancia con un cofre que se encuentra cerrado. Examinamos el grabado con el dibujo de una serpiente que se encuentra en la pared de la sala del enemigo y de repente observamos como éste cobra vida ante nuestros ojos. Si tenemos la suficiente experiencia no nos costará demasiado derrotar a este ser cargando con nuestra espada contra él.

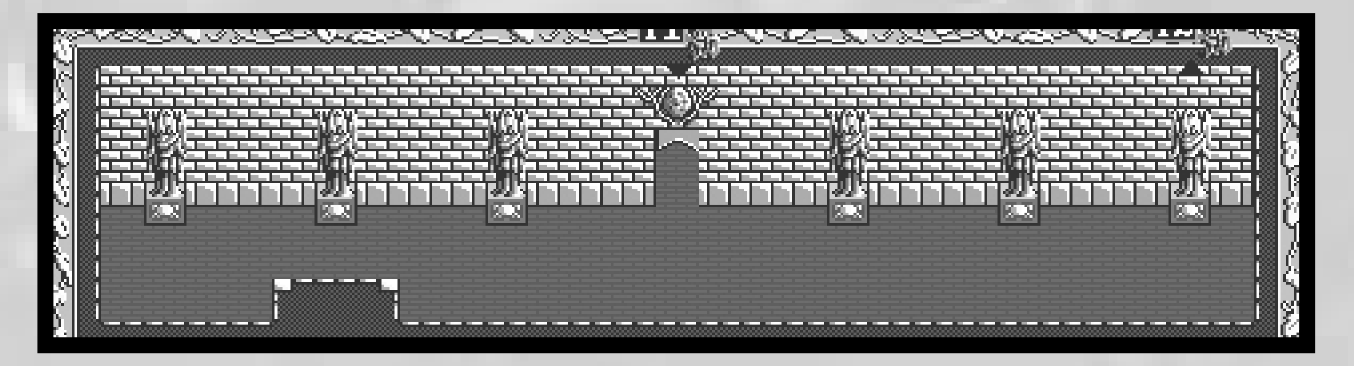
Una vez eliminado, podemos abrir el cofre y obtenemos el **primer libro de *Ys***. Contentos con nuestra victoria, abandonamos el templo.

Continuaremos la aventura de Adol en el próximo número...

Oscar Centelles

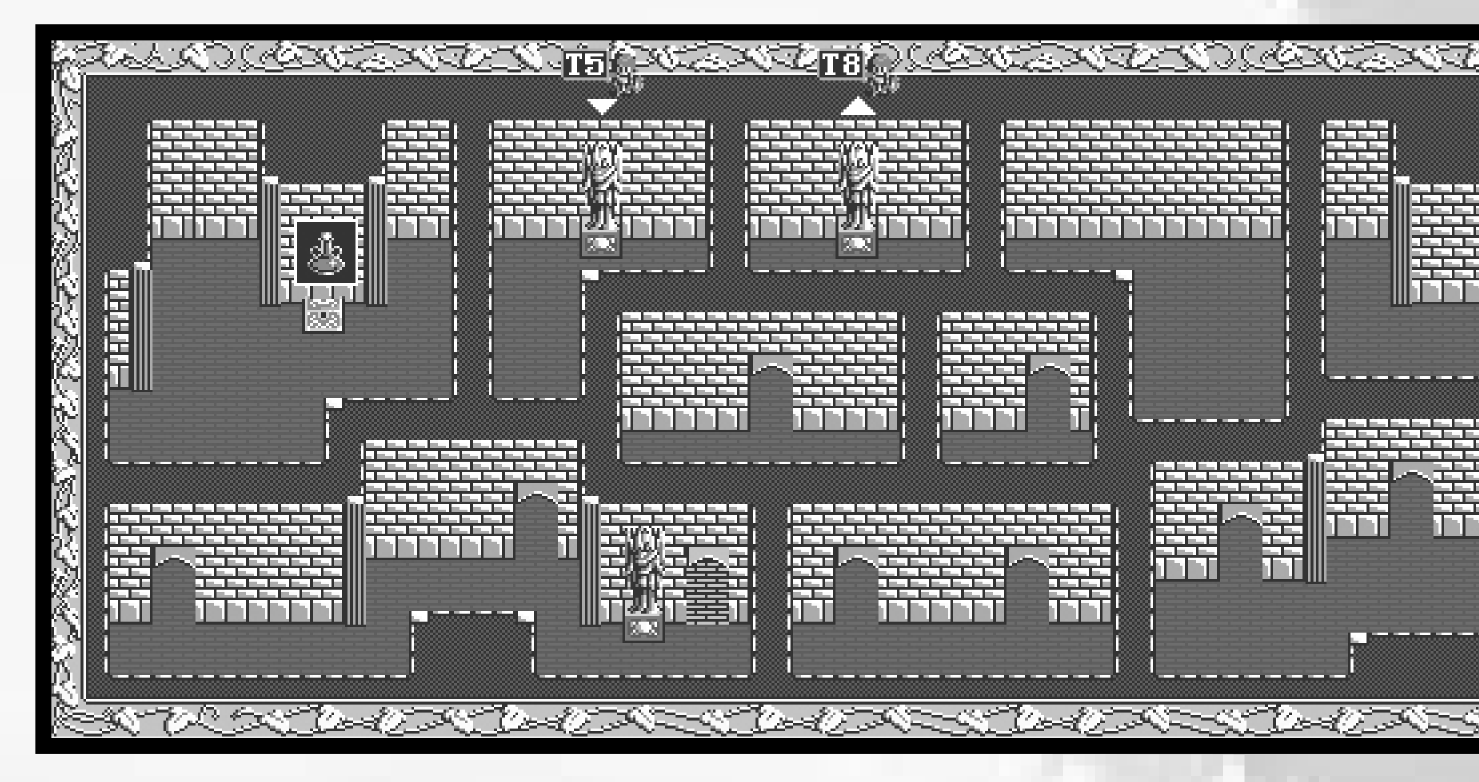
Montaña

Templo 1

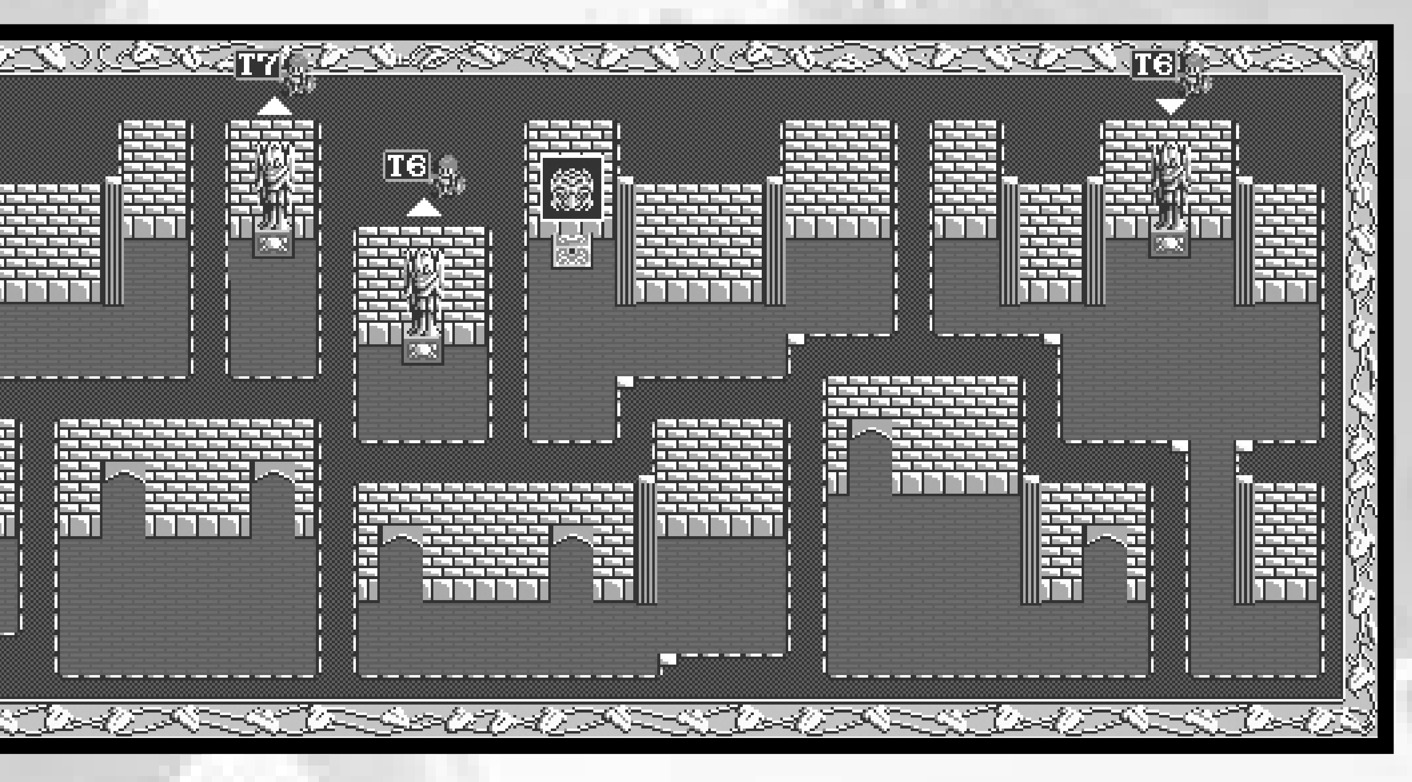


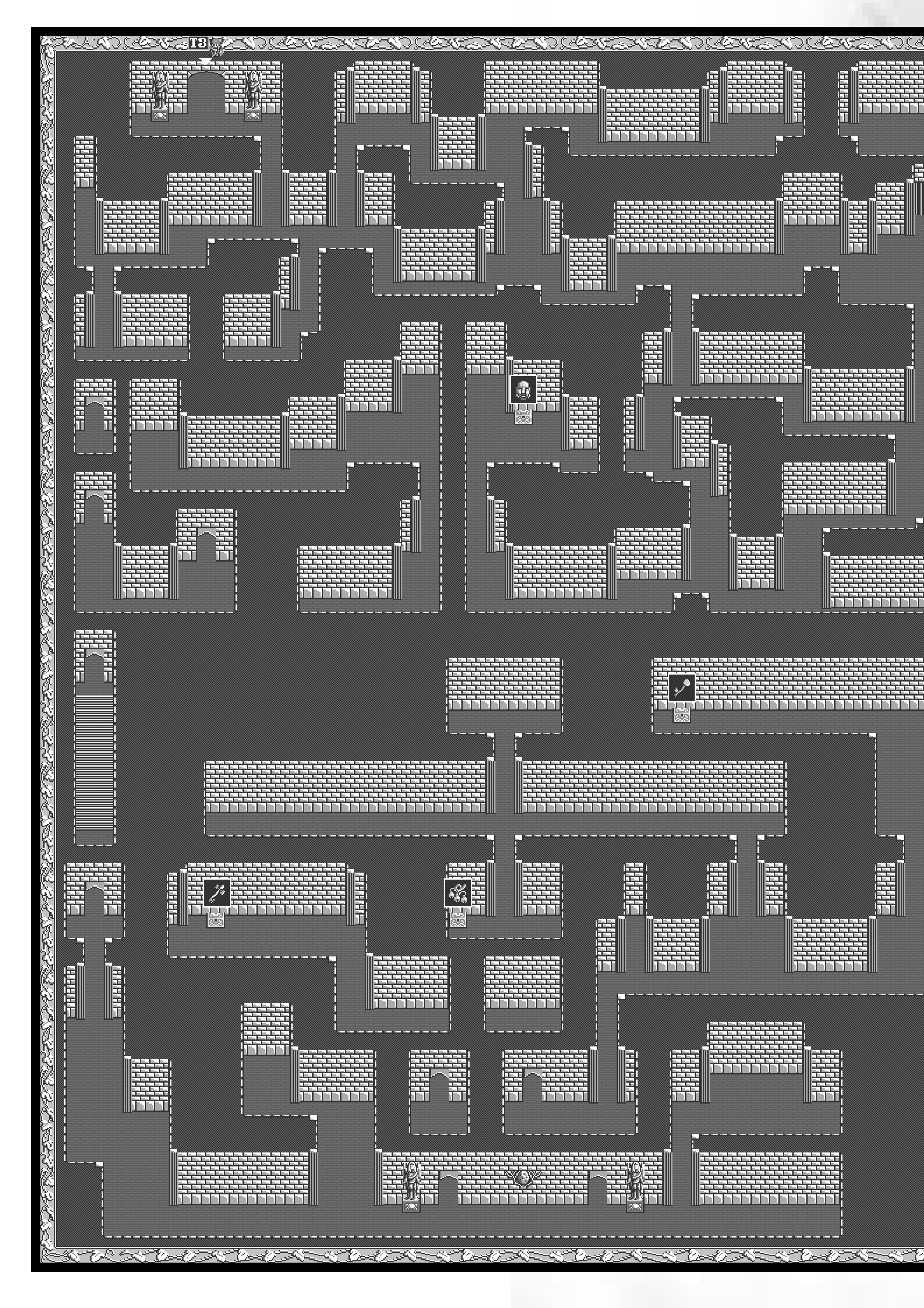
Templo 2

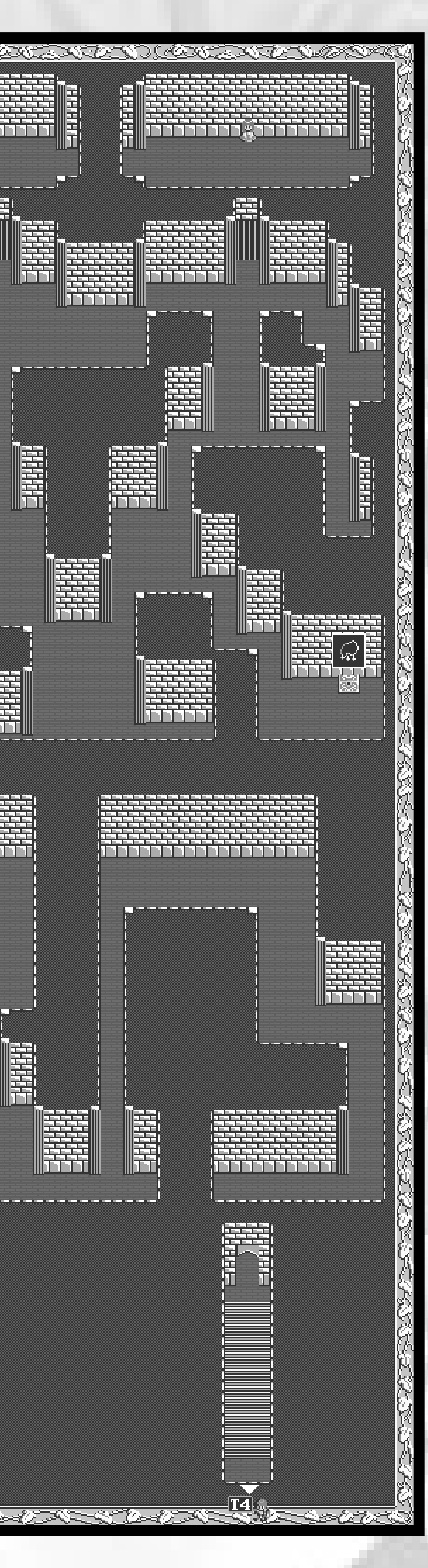
Templo 5



T2



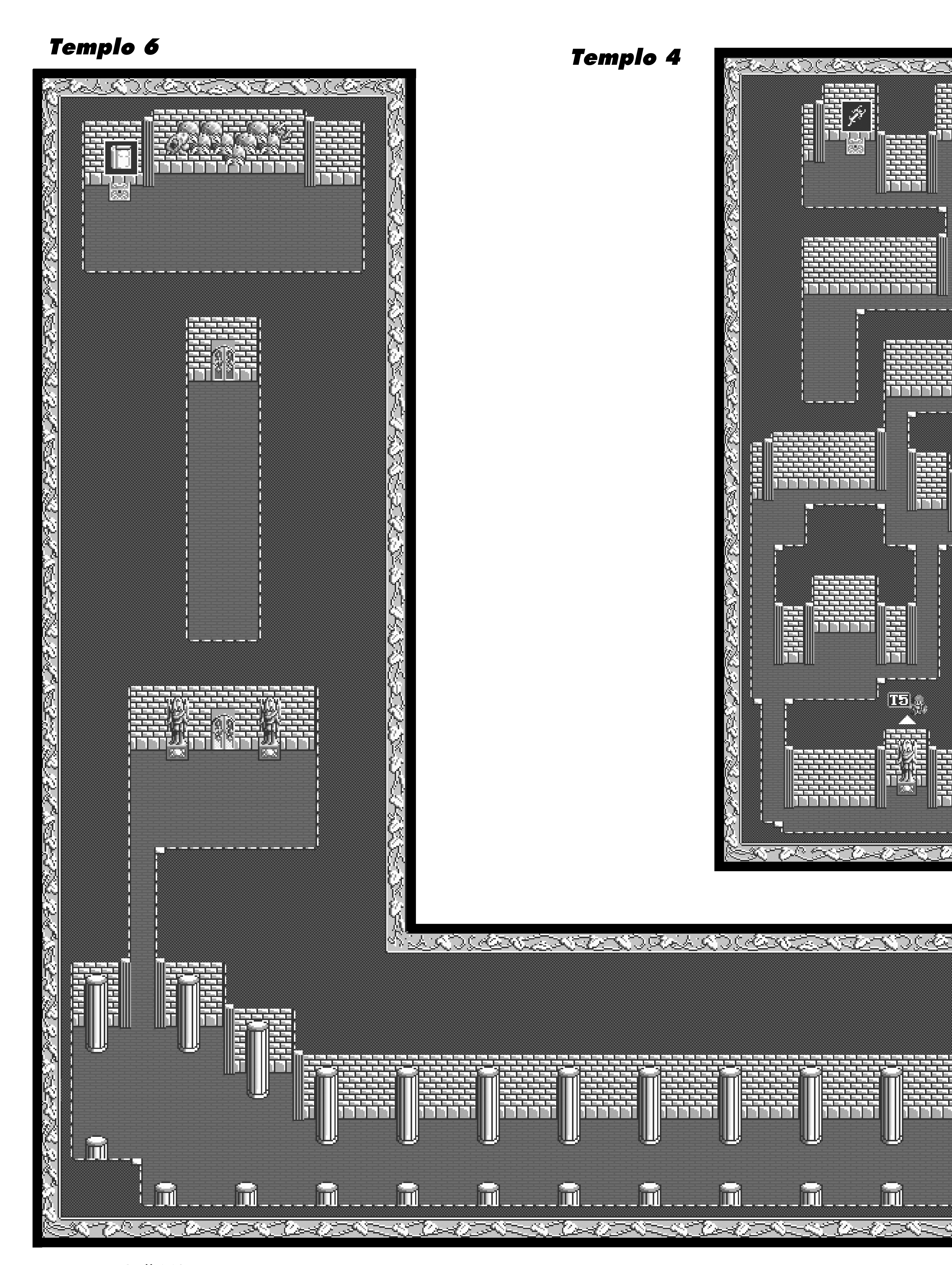

T3



Templo 3

Tabla de enemigos

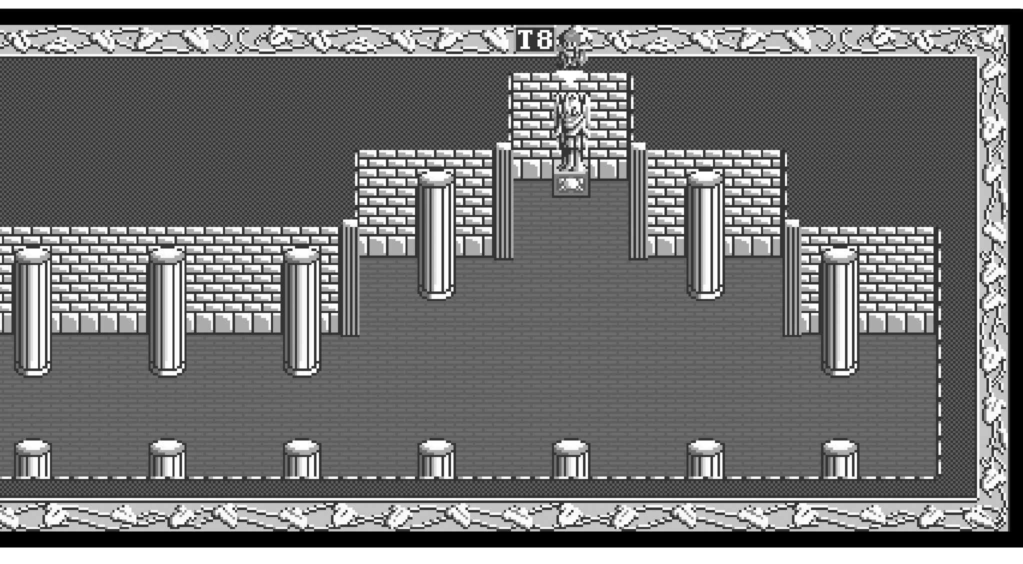
ENEMIGO		EXP	ORO
Oacrot		2	6
Curloyd		2	10
Rheboll		2	14
Urugun		4	14
Hechicero		500	0
Uhnos		40	50
Bolner		50	60
Looter		60	70
Rescoyd		70	100
Lyus		80	120
Dinvel		90	140
Gusano gigante		5000	0


Templo 6
Templo 4
T5

T4

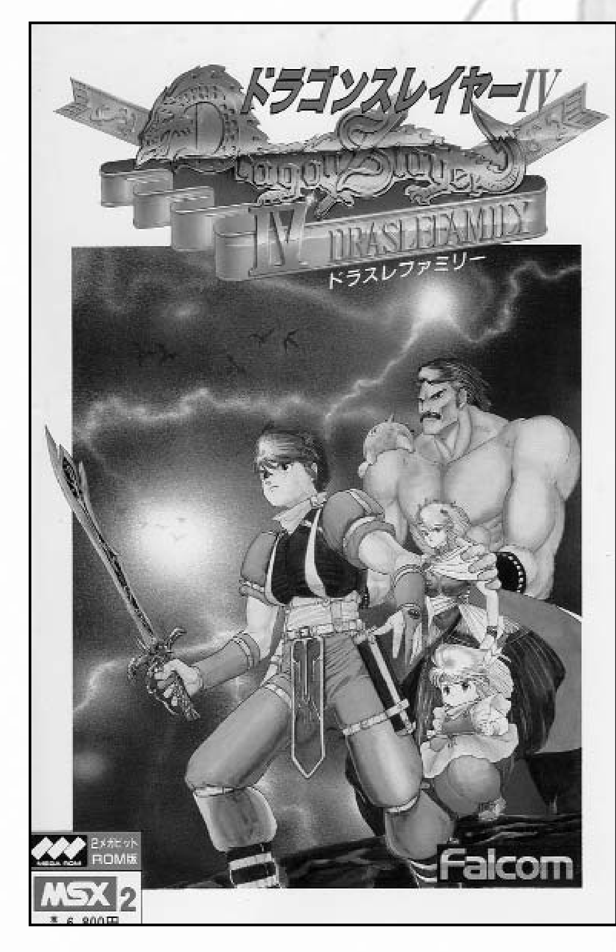
T6

T7



DragonSlayer IV (Drasle Family): 'Cómo Acabar...'

version MSX2



Estamos ante un estupendo juego, aunque un tanto confuso si no se conocen de antemano ciertos aspectos acerca de el.

En cuanto a versiones, aclaremos que originalmente se trata de un titulo de NES (subtitulado en su version americana "The Legacy Of The Wizard"). La version MSX1 es identica estructuralmente (mapeado, localizacion de items, red de 'warps') a la de NES.
La version MSX2, mas cercana graficamente a la version NES, sufre no obstante ciertos cambios estructurales de gran importancia... y esto hace que la escasa informacion que se puede encontrar en inet sobre el juego resulte, en definitiva, bastante inservible (digamos que el juego 'pide ser jugado' de otro modo)..

Aqui va, pues, una ayuda especifica para el doble megarom de segunda generacion.

No quisiera dejar de decir que la version MSX1 es graficamente sorprendente y ademas es algo mas facil que la version MSX2... y que ademas permite usar toda la info de inet (de la version NES) perfectamente.

Bien...
El objetivo del juego es conseguir la espada "DragonSlayer" y usarla para acabar con el Rey Dragon 'Dilguios'. Para ello, antes hay que encontrar 4 coronas (cada una de ellas custodiada por un 'boss').

Contaras con la ayuda de 5 personajes de diferentes caracteristicas y de varios objetos especiales que te permitiran (o haran mas facil) conseguirlo.
El acceso a las diferentes zonas que componen el extenso mapeado esta determinado, de hecho, por esas caracteristicas especiales y tambien por el uso de items concretos. Cuando exploras el juego completo se hace evidente que la estructura esta pensada para jugar de un modo secuencial: A continuacion podras leer el que yo considero el camino mas corto y comodo para disfrutar (y acabar) el juego.

Se me olvidaba... Controles: <cursores> (direcciones), <SHIFT> (salto), <Z> (disparo), <ENTER> (seleccion items).

Y, sobre las claves: selecciona a la abuelita ('Geera Worzen') p ara cogerlas.
Para insertarlas, selecciona al abuelo ('Dawel Worzen').
Y si los kanas no son lo tuyo, existen versiones de las roms adaptadas a teclado europeo. Se acabaron las excusas! Toca jugar......

PRIMERA CORONA (Pochi VS Taratunes).

Selecciona a "Pochi" (Monster). Los enemigos no le restan 'life', y por tanto es ideal para un primer contacto con el juego. Antes de salir se te ofrece la posibilidad de equipar items, pero como no tienes aun ninguno, simplemente sales.
Desde la casa ve hacia la izquierda y baja por la escalera.
Estas en la entrada de la gran mazmorra que conforma el grueso del juego.
A partir de aqui hay 2 caminos basicos: dirigirse hacia la derecha durante un cierto numero de pantallas (un mapeado bastante breve) o bajar y llegar, tras unas pocas pantallas (abajo, izquierda, abajo, derecha, abajo), al punto de partida de los 4 grandes recorridos... el lugar donde se encuentra el dragon.
Ve donde el dragon. En tu camino dispara a algunos enemigos para conseguir llaves (procura no desperdiciar magia) y abre un cofre que contiene una 'bola-warp' (sirve para volver a la superficie).
Desde el dragon, a la derecha y por la escalera hacia abajo.

Siempre hacia la derecha, descubriras que llegas a una zona sin salida. En ella encuentras un cofre que contiene una 'pocion' (sirve para recargar magia) y una tienda. Es importante recordar esta zona, ya que bajando por la escalera junto a dicha tienda llegas a una pantalla donde se encuentra un 'warp' (la imagen de una chica de perfil) muy importante. Estos warps solo pueden usarse cuando vas consiguiendo las coronas: los usaremos mas tarde, a mitad y a final de juego.

Vuelve hacia la izquierda.
En la misma pantalla por donde entraste hay una escalera que baja. Abajo pues y sigue hacia la izquierda.
Entras en una zona con muchas

escaleras. Has de encontrar el camino para bajar por una escalera hacia la siguiente area.

Al bajar entras en una zona de bloques blancos. Hacia la derecha, usa a los lentos enemigos voladores para alcanzar la parte superior de la pantalla (subete



encima de ellos y salta) y pasa a traves de un 'falso bloque'. Durante el juego es muy comun la existencia de estos falsos bloques que se destruyen por contacto.
Abre el cofre: encuentras un valioso item, los 'nudillos' (cuadruplican el poder de ataque). Sigue hacia la derecha y baja por las escaleras al final de esta zona.

De nuevo montones de escaleras (bloques verdes esta vez en lugar de los ladrillos marrones de antes).
Hay que encontrar la salida de esta zona, hacia la derecha.

Una vez hayas salido, lo primero que ves es una posada ('INN'). Si lo necesitas, entra y gasta 10G para reponer vida y magia. Tambien puedes equipar items, aunque no es necesario por ahora.
Tambien hay una tienda en esta zona.
Y lo mas importante: un cofre (que contiene una gran bolsa de 'gold') en la parte superior de la pantalla, muy cerca del techo. Tras coger el cofre, si saltas, abriras un pasaje secreto (falso bloque-techo) hacia la zona donde se esconde la primera corona.

PERO ANTES, es necesario explorar un poco mas (vamos a buscar otro valioso item escondido en un cofre cerca de aqui).
Ve hacia la derecha y baja por la escalera que hay al extremo.

Dirigete hacia el extremo izquierdo, baja... y de nuevo hacia el extremo derecho (no es necesario entrar hacia donde se encuentran la posada y la tienda).
Baja por la escalera.

Cuidado con los pinchos en esta zona. Existe un truco para no perder tanta vida, y consiste en mantener pulsado <ARRIBA> al tiempo que avanzas hacia izquierda o derecha.
Escondido entre los pinchos, encuentras el 'pico' (item que solo puede ser usado por 'Lyll', y que sera fundamental para conseguir la segunda corona, sirve para romper bloques).

Ya puedes volver a la pantalla del pasaje secreto. A por la corona.

Sube por el hueco tras romper el falso techo.
De nuevo muchas escaleras. Has de llegar al extremo derecho del area (investiga bien, hay unos cuantos falsos bloques-ladrillo que se rompen al tocarlos). Procura, ademas (y como siempre) llevar algunas llaves. Te haran falta.
Veras, a la derecha, una posada y una tienda. Sobre la tienda, de nuevo falso techo y pasadizo, ya si, a la pantalla donde se encuentra el cofre que esconde la corona (un lugar repleto de falsos bloques-ladrillo azules). El cofre que te interesa se encuentra justo encima de la posada (por cierto seria un buen momento para recargar vida y magia).

Abre el cofre, coge la corona... y derrota a 'Taratunes'. Es facil: Pochi es fuerte y en pocos golpes Taratunes sera historia (pero ojo, que esta gran bestia si te resta vida, a diferencia de los demas enemigos normales del juego)..
Conseguida la corona, eres teletransportado a la superficie.
A casa: antes de seguir, coge la clave.

SEGUNDA CORONA (Lyll VS Erebone).

Selecciona a 'Lyll' (Elf).
Y antes de salir no olvides equipar los items 'pico' y 'nudillos'.
Procura llevar los nudillos seleccionados para matar a los enemigos con menos golpes (y ojo con el marcador de vida, que el unico personaje invulnerable es Pochi). Atento tambien al marcador de magia: disparar con los nudillos seleccionados duplica la magia que se gasta.

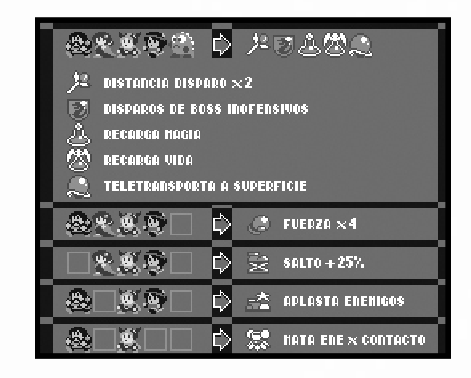
Antes de ir en busca de la corona, vamos a localizar un item muy importante (las 'botas de pinchos', con las cuales podras aplastar a los enemigos saltando sobre ellos y sin gastar nada de magia).

Ve donde el dragon.
Hacia la izquierda y sube por la escalera (hasta que no puedas subir mas).
Ahora, a la izquierda.
Llegas a una zona donde hay muchos bloques blancos (que podras romper con el pico). Desde la posada hacia la izquierda (siempre en contacto con el techo) busca un falso bloque-techo en el pasillo donde esta el pulpo. Salta y entra.
Seguro que reconoces esta pantalla: pasas por aqui siempre que vas hacia el dragon. Abre el cofre: ya tienes las botas de pinchos. Vuelve a la posada y descansa.... antes de salir, equipa las botas. A partir de ahora casi no gastaras magia y el juego sera un poco menos duro.

De vuelta al punto de partida (el dragon).
Hacia la derecha. Hay que subir (por la escalera de la derecha: la de la izquierda sirve para volver a la superficie... ten esto en cuenta para no gastar bolas-warp).

En esta zona hay una bola-warp en un cofre. Buscala antes de dirigirte hacia la derecha.

A golpe de pico, hacia la derecha, llegas a la siguiente zona, con motivos graficos "vegetales".



No te queda mas remedio que bajar (hasta el fondo, usando el pico cuando sea necesario). En el fondo encuentras un 'cetro' (duplica la longitud del disparo) escondido en un cofre.
Ahora hay que subir a la misma pantalla donde comenzo todo, pero quedando en su lado derecho (puedes coger el cofre, que contiene una pocion para recargar magia).

Salta hacia la derecha y camina hacia la siguiente pantalla.
Dejate caer por el enorme precipicio.
En el fondo, una tienda: compra las ‘botas de salto’ (95G). Equipalas entrando en una posada y vuelve a subir (usando botas de salto y pico cuando sean necesarios).... si, de nuevo a la primera pantalla de esta zona.

Gracias a las botas de salto y desde esa primera pantalla, podras acceder (arriba y a la derecha) a la zona donde se encuentra la espada DragonSlayer (que no, aun no puedes coger).
Desde ahi, arriba y hacia la izquierda... llegaras asi a una nueva zona de bloques blancos (los enemigos, unas adorables chicas-gato).

Sigue hacia la izquierda, hasta llegar a una zona con bloques en tonos rosa.
Recuerda este lugar. Sera importante al final del juego.

Siempre con mucho cuidado, saltando y hacia el extremo izquierdo.... hasta llegar a otra pantalla que te resultara familiar: ves un cofre y un leon inmovil (tambien pasas por aqui siempre que vas hacia el dragon).
Ese cofre contiene la segunda corona. Antes de cogerlo, entra en la posada y equipa antes de salir: las botas de salto y los nudillos.

Ahora abre el cofre (deberas tocarlo por el lado izquierdo, ya que el derecho esta bloquedado por un bloque invisible) y enfrentate a ‘Erebone’. El gran salto de Lyll te lo pondra facil.

Conseguida la segunda corona, eres teletransportado a superficie.
(de nuevo buen momento para coger la clave).

“INTERMEDIO”: Recogida de Items especiales.

Antes de salir en busca de la tercera corona, es necesario obtener ciertos items quenos haran falta mas adelante.

Selecciona a ‘Royas’ (Ranger). Equipale una de las coronas, las botas de salto y las botas de pinchos.
Desde el comienzo del mapeado, vamos por fin a ver que hay en ese corto mapeado a la derecha de la entrada... Es importante, mientras caminas hacia la derecha, que te entretengas en aplastar con las botas de pinchos el mayor numero posible de enemigos. De este modo rellenas el contador de oro y tambien vas llenando el de llaves.

Has de llegar al extremo derecho de la zona (tendras que buscar bloques falsos y romperlos por contacto, y a veces usar las botas de salto).
Baja por la escalera y coge ese cofre: consigues el ‘guante’ (item que tan solo podra usar el guerrero ‘Xemn’ en la busqueda de la cuarta corona, y que sirve para mover bloques).

Vuelve al punto de partida, siempre



recolectando oro y llaves. Llena el marcador de oro AL MAXIMO (vas a necesitar 95G en un momento).

Dirigete donde el dragon.
Desde aqui, ve hacia la izquierda o hacia la derecha, da igual. Ponte bajo cualquiera de los 4 warps que hay en esta zona... equipa la corona y pulsa <ARRIBA>.
Seras teletransportado a una tienda muy especial. Entra y compra el ‘escudo’, el unico que encontraras en todo el juego, y que a partir de ahora te permitira enfrentarte a los bosses sin que sus disparos te quiten vida. En esta tienda tambien se venden elixires (que te recargan al completo la vida y te resucitan al morir) a muy buen precio (10G).

Para salir de aqui, salta: el letrero de la tienda se transformara en un warp.
De nuevo corona y <ARRIBA>. Apareces donde el dragon.

(Aclaro que no toda la red de ‘warps’ se activa al coger la primera corona. Es necesario tener al menos 2 coronas para que se activen los ‘warps’ que te llevan a la tienda del escudo... y, claro, hasta 4 coronas -el maximo- para que se activen todos y poder acceder asi a la escondida espada legendaria)

Vuelve al casa. Para hacerlo, dirigete hacia la derecha y toma la escalera superior izquierda.... etc. Podrias usar una bola-warp, pero es conveniente recolectar durante el camino oro y llaves (usando las botas de pinchos).

En la casa, selecciona a Lyll. Equipale las botas de salto, las botas de pinchos y el pico. Vamos a por otro item importante (el ultimo antes de continuar con la busqueda de coronas).

Hacia el dragon, de nuevo. Recolecta mucho oro (necesitaras 95G enseguida).
Desde el dragon, de nuevo a la derecha y arriba (escalera superior derecha). En la segunda pantalla a la derecha, sube por el hueco en el techo, y hacia la derecha.
Usando las botas de salto y el pico, abrete camino hacia la tienda que hay a la derecha: compra las ‘alas’ (95G), un item que solo puede ser usado por la maga ‘Maia’ en la busqueda de la tercera corona.

Y vuelta a casa... aunque es aconsejable, antes de volver, dedicar unos minutos en esa comoda zona a la derecha de la entrada para (siempre gracias a las botas de pinchos) recolectar el maximo de oro y todas las llaves posibles.
En la busqueda de la tercera corona sera necesario llevar el marcador de oro al maximo y, al menos, la tercera parte del marcador de llaves lleno.
Estamos, en este momento, en la mitad aproximada del juego.
(no estaria mal coger una clave antes de seguir.....)

TERCERA CORONA (Maia VS Archwinger).

Selecciona a ‘Maia’ (Wizard). Equipale las alas, los nudillos y las botas de salto. Lastima que no pueda usar las botas de pinchos (cuidado ahora: acostumbrado a usar aquellas ahora tendras que usar los nudillos y controlar de nuevo los

marcadores de vida y magia).

Hacia el dragon, a la izquierda, baja por las escaleras.
Caida libre y veras la tienda donde se compra el escudo.
Baja por las escaleras y despues todo recto hacia la derecha y luego otro buen trecho de escaleras que bajan.

Llegas a una zona de bloques naranjas.
Dirigete hacia la izquierda. Busca bloques falsos para no tener que enfrentarte a los pinchos y enemigos y ahorrar llaves.

Siempre hacia la izquierda, pasada la pantalla del castillo, llegas a una zona que te da a elegir 3 caminos: arriba, centro, abajo. Por el camino central podras acceder a un cofre que contiene una bola-warp.
Para continuar la busqueda de la corona, el camino es ABAJO. Hasta la escalera. Baja.

Zona de bloques amarillos. A la derecha hay una tienda: gasta 90G en el 'arco', un item que solo puede ser usado por Maia y que sirve para empujar los mismos bloques que Lyll puede romper con el pico y que Xemn podra mover con el guante.
En el extremo derecho hay una posada y bajo esta un cofre que contiene una pocion para recargar magia. Descansa en la posada y equipa el arco.

Vuelve al extremo izquierdo de la zona: el cuarto bloque del suelo empezando por la izquierda es falso... pulsando <ABAJO> lo rompes y accedes asi a una nueva zona (con un bonito fondo de color azul).

Hacia la extremo derecho y sube.

En esta nueva zona, en el extremo izquierdo hay unas escaleras que conducen a la posada que has visto hace poco (por cierto se convierte en tienda si saltas sobre el letrero).
Pero lo que nos interesa es ir arriba y hacia la derecha, hasta dar con un hueco en el suelo por el que te dejas caer.

Estas en una zona verde-azul.
Ahora toca abrirse camino entre un monton de falsos bloques para salir por la parte inferior derecha de la pantalla. Una vez fuera, y en el extremo derecho de la zona, encuentras un cofre con una gran bolsa de oro (tambien hay una tienda).

Arriba.

Hacia la derecha, ahora. Hay que pasar una larga hilera de puertas. Gastaras muchas llaves (usando el arco y buscando bloques falsos puedes ahorrar algunas).

Llegas a una nueva zona con bloques azules y chicas-gato.
Hacia las escaleras, para bajar a la siguiente zona (fondo azul y rocas marrones).

De nuevo a la derecha, pasas una posada (convertible en tienda si saltas sobre el letrero). Si no tienes las alas equipadas, tendras que entrar en la posada, descansar y equiparlas: hay que hacer uso de ellas a continuacion.
Llegas, a la derecha, hasta un cofre que contiene 20 llaves. Sube.

Hacia la izquierda, gastando llaves.
Ahora tocara romper algunos bloques y subir por las escaleras, llegando a una nueva zona (de bloques marrones).

Dirigete hacia la derecha (cuidado con los pinchos), sube... y de nuevo hacia la izquierda. Por fin ves una tienda envuelta en una espiral de bloques. Llega hasta ella (usando las alas) y compra (90G) la 'llave magica', un item que solo puede usar Maia y que permite abrir puertas gastando magia en lugar de llaves (menos mal, porque seguro que ya te quedaban pocas).

Continua hacia la izquierda y sube: zona de ladrillos azules y verdes, de nuevo con muchos pinchos. Hay que arreglarselas para subir.
en el extremo izquierdo de la zona hay un cofre que esconde un anillo (que te hara invulnerable un ratito).
En el extremo derecho hay una posada: entra, descansa y equipa la llave magica antes de salir (tambien las alas y el arco).

Sube.
A la izquierda y de nuevo baja (escaleras).

Usando primero el arco para empujar el bloque marron y despues la llave magica para abrir todas las puertas, abrete camino hacia la derecha (ojo a los enemigos "invisibles").

Estamos ya cerca de la corona...

Ves una posada (no entres aun, a no ser que estes fatal de vida y/o magia).
A la derecha, baja y usa la llave magica para abrir todas las puertas hacia la izquierda. Hasta que veas el cofre (no, no lo cojas aun!).

Vuelve ahora a la posada (siempre cuidado con los enemigos "invisibles").
Entra, descansa y equipa: alas, escudo y un elixir.

Ahora si, a por el cofre.
Abrelo y enfrentate a 'Archwinger'. Gracias al escudo sus disparos no seran un problema. Cuidado con los golpes fisicos (usa el elixir si te queda poca vida).

Derrotada la bestia, volvemos a casa.
Ha sido una larga busqueda, no olvides tomar la clave....

CUARTA CORONA (Xemn VS Rockgaea).

Selecciona a 'Xemn' (Warrior). Equipale las botas de pinchos y el guante.

Ve donde el dragon, y dirigete a la izquierda. Sube por la escalera.
Arriba del todo, coge ese cofre (contiene una gran bolsa de oro). A la derecha ahora.

Estas en la misma zona donde antes vinimos con Lyll para coger las botas de pinchos.

Ahora has de aprender a usar el 'guante', un item que te permite mover ciertos bloques.
Es necesario hacerse con su funcionamiento (que al principio puede parecer terriblemente confuso, pero en realidad es francamente sencillo).
Basicamente, consiste en tocar bloques con <SHIFT>+<direccion> pulsados. De este modo el bloque se movera en dicha direccion (si ningun obstaculo lo impide, claro).
Para ciertos movimientos no es necesario mantener pulsadas las teclas de <direccion>: tan solo manteniendo pulsado <SHIFT>, y al contacto con el bloque, este se movera EN LA ULTIMA DIRECCION EN QUE QUE TU TE MOVISTE.
Esto te sera muy util: por ejemplo para mover hacia la izquierda un bloque que esta sobre tu cabeza tan solo has de situ-

arte debajo (viniendo desde la derecha, para que la ultima direccion pulsada sea <izquierda>). A continuacion, tan solo saltando y manteniendo pulsado el salto durante el contacto con el bloque, este se movera hacia la izquierda.
Y otro ejemplo: en pantallas posteriores tendras que mover columnas completas de bloques hacia la derecha. Nada mas facil: salta sobre el primer bloque, pulsa <derecha> en mitad de salto (y suelta-lo)... al caer sobre el bloque (y siempre manteniendo pulsado <SHIFT>) este se desplazara hacia la derecha. Si mantienes la tecla pulsada, todos los bloques de la columna se moveran hacia la derecha de un modo facil y rapido (uno tras otro conforme vayas apartandolos y cayend o consecutivamente sobre el siguiente).



Tambien tendras que atravesar algunos precipicios moviendote sobre un bloque al tiempo que mueves el susodicho: en ese caso, bastara con situarse al borde del bloque (en la direccion deseada) y saltar, manteniendo pulsado <SHIFT> al caer. Con ello, el bloque se movera en esa direccion y tu permaneceras aun montado encima. Repetir cuantas veces sea necesario.
Con un poco de practica conseguiras incluso mover bloques en diagonal en mitad del salto...

En fin, si has aprendido a usar el guante habras conseguido avanzar hacia la izquierda (hasta el extremo inferior izquierdo), donde hay una tienda. Junto a ella, un falso bloque-suelo. Rompelo y dejate caer.

Estas ahora en una zona de ladrillos naranjas y fondo azul.
Hay que dirigirse hacia la derecha, haciendo mas uso del guante. En el extremo derecho de la zona, abajo, de nuevo un falso bloque-suelo. Romper y bajar.

Entras en una zona de grandes caidas. Cuidado.
Siempre por la zona de arriba, avanza hacia la izquierda (hay algunos falsos bloques-suelo, atencion). Tendras que dejarte caer por un hueco (aprovecha para coger el contenido del cofre, una pocion magica) y vuelve a subir (escalera) para continuar hacia la izquierda.

Llegas a una pantalla con bloques azules y muchos pinchos.
Usa el guante para mover los bloques, has de bajar por el hueco que hay abajo a la izquierda.

Nueva zona y uso masivo de guante, unido ademas a unas ciertas dosis de 'puzzle'(largas horas dedicadas a 'Eggerland Mystery' dan, por fin, sus frutos ;D).
El objetivo es llegar al extremo superior derecho de la zona, romper unos bloques falsos y dejarse caer por el falso suelo (extremo inferior derecho).

Mas trabajo para el guante, ahora avan-zando hacia la izquierda.
Entra en el hueco donde se encuentran la posada y la tienda. Salta en la tienda y coge el cofre: consigues una 'armadura'. Entra en la posada, descansa y equipa la armadura: los enemigos moriran cuando los toques (ojo, gasta mucha magia y de modo constante mientras la llevas pues-ta... usala solo en pasajes estrechos cuando no puedas saltar para utilizar las botas de pinchos).

Ve a la izquierda y baja por la escalera.

Zona gris-azul. Hay que ir al extremo derecho de la zona y bajar por la escalera.

Llegas asi a una zona en la cual deberas usar (y abusar) del guante y quiza un poco la armadura. Vuelve a leer el parrafo donde explico como mover columnas enteras de bloques con el guante (te hara falta).
En fin, el objetivo de esta zona es llegar al extremo inferior izquierdo (donde hay una tienda y una posada). Baja por la escalera izquierda.

MUCHO CUIDADO AHORA: has de diri-girte hacia la derecha, pero siempre procurando mantenerte en la parte alta de la pantalla, usando para ello los guantes y los bloques. Procura no caer al suelo, lo cual significaria repetir una buena parte del mapeado. En el extremo derecho de la zona, baja por la escalera.
(nota: si te caes al suelo, puedes intentar volver a subir al bloque desde el cual te has caido usando a los enemigos como trampolin.... pero esto requiere paciencia y puede gastar mucha vida; es casi mas sencillo volver a repetir parte del recorri-do)

Llegas a una zona con pinchos. Hay que llegar a la salida inferior derecha, para lo cual habra que dar un rodeo (izquierda, bajar, derecha).... ojo a los falsos bloques y al uso del guante.

POR FIN
Nos acercamos a la corona.

Hacia la derecha, sube las escaleras (puedes coger ese cofre, contiene un elixir) y hacia la izquierda (hay que entrar en el hueco donde se encuentra la posa-da).

Sube las escaleras hasta arriba del todo. Esta zona debe resultarte familar: es donde comenzaba la busqueda de Maia.

Investiga los bloques falsos, avanzando hacia la derecha y, ojo, procurando no caer abajo (habria que volver a empezar). Asi hasta llegar a la posada. Entra, descansa y equipa (al menos): escudo y pocion magica.

Hacia la derecha, ahora si busca un falso bloque-suelo y dejate caer.

Bien: ESE es el cofre que contiene la cuarta corona.
Para abrirlo, tendras que saltar cuando haya un enemigo a su derecha ("agar-rarte" a el en mitad de salto y avanzar hazia la izquierda hasta tocar el cofre).

La lucha contra 'Rockgaea' es dificil. El escudo te hace invulnerable a sus disparos, pero su contacto te resta gran cantidad de energia.
Por eso, en lugar de equipar un elixir y lanzarte a una batalla con pocas posibili-dades, es mejor armarse de paciencia y dispararle desde una posicion segura (desde arriba, disparandole en diagonal descendente). Necesitaras bastante magia (de ahi equipar tambien una pocion magica, por si se te acaban los puntos

magicos).

Derrotado Rockgaea y obtenida la cuarta y ultima corona, a casa.
Ten buen cuidado de coger la clave. Posiblemente sea la ultima que uses. (ademas de que ha sido un poquito dificil de conseguir, con tanto trabajo de guante y demas, verdad?)

BATALLA FINAL (Royas VS Dilguios, el Rey Dragon).

El ultimo tramo del juego consiste en la busqueda de la espada DragonSlayer.
Para ello es obligatorio usar a 'Royas' (Ranger), el unico que puede usar las coronas para teletransportarse gracias a la red de 'warps'.

(nota: en esta version MSX2, dicha red es mas amplia que en las versiones NES y MSX1... y ademas los diferentes 'warps' estan interconectados de otro modo)

Tambien sera obligatorio usar a Royas en el enfrentamiento final, pues es el unico personaje capaz de usar la espada legendaria.
Afortunadamente, en la busqueda de la espada podremos usar las botas de pinchos, la armadura y las botas de salto (Royas es un personaje bastante debil en general).

Equipa, pues: corona, botas de salto y botas de pinchos (o armadura).

Ve donde el dragon.

Ahora, a la derecha y baja por la escalera. Esta es la zona donde comenzaba la busqueda de Pochi.
Todo recto hacia la derecha (hasta el final, veras una tienda -junto a la que cogiste, recuerdas?, una pocion magica contenida en un cofre), y despues baja por la escalera: este es el 'warp' con el que se inicia el camino. Selecciona la corona y pulsas <ARRIBA>.

Apareces sobre la espada, pero aun no puedes cogerla.
A la derecha, salta y dejate caer hasta el fondo.

Ahora hacia la izquierda (usando botas de salto y tambien apoyandote, en ocasiones, sobre algun enemigo), hasta llegar al siguiente 'warp' (situado sobre un monton de bloques amarillos).

Corona + <ARRIBA> (="warp!").

Reconoceras esta zona por estar muy cerca del tercer boss (corona obtenida por Maia).
Avanza hacia la izquierda gastando algunas llaves.
warp!

Con cuidado, salta hacia la derecha. Abre la puerta, salta con las botas...
warp!

Zona gris: baja hasta el final.
Ahora un poquito hacia la derecha y...
warp! (PERO OJO: ahora es necesario mantener pulsado, tras el warp, la tecla <IZQUIERDA>, solo un poco, para caer sobre una columna).

Si te has caido al fondo, deberas volver a empezar (si, todo, desde el principio).
Si en cambio estas sobre la columna rosa, salta con cuidado hacia la izquierda. (esta zona rosa ya la vimos mientras buscabamos la segunda corona con Lyll)
warp!



Bien... ya puedes coger la espada DragonSlayer.
IMPORTANTE: desde el momento que coges la espada, las claves que te de la abuela Geera seran inutiles. En otras palabras, para ver el final del juego hay que hacer necesariamente la busqueda de la espada y enfrentarse al dragon (todo en una sentada, no queda remedio).

Usa el 'warp' que hay sobre la espada.

Ahora avanza hacia la derecha (ingnora ese 'warp' que ves en la pantalla en la que apareces) sube, gasta 4 llaves.... y, ahora si, toma ese otro 'warp' que ves.
Te lleva a una posada.
Entra, descansa y equipa una bola-warp.
Sal, usala. A casa!

Insisto, no te molestes en tomar clave.

Selecciona de nuevo a Royas y equipale las botas de pinchos (o armadura) y la espada.
Ve donde el dragon.
Puedes descansar en la posada que hay sobre el, y equipar la espada ahora si no lo habias hecho antes.
Baja, situate bajo el.... y por fin se producira el enfrentamiento final contra Dilguios, el Rey Dragon.

Tan solo necesitas un poco de habilidad y paciencia: disparando rafagas y saltando hacia la izquierda le vas restando vida y evitando su aliento de fuego.
Cuando ambos llegais a la parte izquierda de la pantalla, Dilguios retrocedera volando y vuelta a empezar. Como digo, tan solo un poco de habilidad y paciencia.

INCISO: en la version MSX1 el dragon debe ser derrotado de un modo diferente...
Tras cada "rafaga de disparos", has de retroceder *completamente* hasta el extremo izquiedo de la pantalla: solo de este modo el bicho saltara retrocediendo hacia la derecha.
En caso contrario, avanzaria hasta arrinconarte (a la izquierda)... abrasandote sin remedio.
De modo que toca tener un poco mas de paciencia que en la version MSX2... pero dado que el juego en general es algo mas facil para primera generacion, bien esta que el dragon sea un poco mas dificil... no?

Fin del dragon, fin del juego.

Durante el epilogo podras disfrutar aun de un par de maravillosas melodias, al mismo nivel que el resto de la banda sonora. En los creditos, una explicacion en 2 palabras (YUZO KOSHIRO) al despliegue de canciones increibles que podemos escuchar a lo largo del juego.
:)

Espero que hayais disfrutado tanto como yo jugando a DragonSlayerIV.

Sut. < sutchan@wanadoo.es >



Antartic Adventure
Konami 1984
ROM 16Kb
RC701

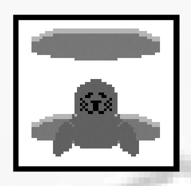
Antartic Adventure fue la segunda aventura de Konami para MSX. Una vez más se presenta en formato cartucho, con 16 kb de memoria.
Esta vez debemos ayudar a un pequeño pingüino durante su viaje por todo el helado continente. Para llevar a cabo a cabo este viaje disponemos de un tiempo límite, así que deberemos ser habilidosos para llegar a buen fin. He aquí el mapa completo que muestra el largo camino y las diferentes etapas:



Obstáculos

Como todo buen juego, lógicamente hay multitud de obstáculos que se opondrán a la marcha de nuestro pequeño amigo. Aquí tenemos una relación de todos ellos:

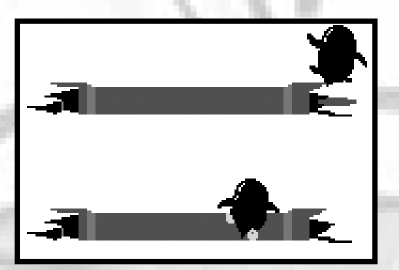
Charcos y focas



Estos pequeños charcos se encuentran repartidos por todo el continente. Son fáciles de esquivar o saltar, pero a veces guardan alguna sorpresa; en algunos de ellos encontraremos simpáticas focas dándose un chapuzón, y como no vayamos con cuidado e intentemos saltar el charco, acabaremos pegándonos un coscorrón contra estos pobres animales. Es mejor intentar esquivar el charco siempre para evitar estos casos. Siempre es posible saltar por encima de la foca si esquivamos su cabeza.

En todo caso tanto si tropezamos con el charco como si nos damos de bruces con la foca, perderemos parte de nuestro tiempo intentando recuperar la marcha.

Grietas en el hielo

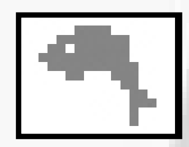


Hay zonas del hielo que están agrietadas y que representan un peligro para los pingüinos despistados. Si topamos con este obstáculo por los laterales, tropezaremos y perderemos algo de tiempo; en cambio, si caemos de pleno dentro de ellos, tendremos que esforzarnos para poder salir de él rápidamente; para ello deberemos pulsar la tecla del cursor hacia arriba y la barra espaciadora a la vez. La mejor manera de sortear este obstáculo es saltar por encima de él (tecla espaciadora) o esquivarlo por los lados (aunque suele ser más difícil debido a la longitud del mismo).

Puntos

Durante la marcha también encontraremos cosas buenas que nos aumentarán nuestra puntuación. Son las siguientes:

Pescado fresco - ***300 puntos***



Hay que ser habilidoso para ayudar a nuestro pingüino a cazar este sabroso alimento. Los pescados saltarán desde los helados charcos para luego caer; hay que estar atentos para interceptar la caída o nuestro amigo pasará hambre.

Banderilla verde - ***500 puntos***



Encontraremos muchas de estas clavadas en nuestro camino; estas son la huella dejada por el hombre en este remoto lugar. Hay que estar atento y no dejar pasar ni una.

Las etapas

El juego está dividido en diez etapas que dan la vuelta a toda la Antártida; al final de cada una de ellas nuestro pingüino podrá tomarse algo calentito en los refugios que diferentes expediciones de varios países habrán instalado allí.

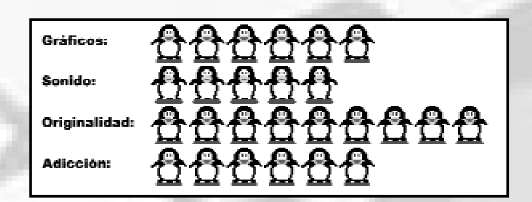
El juego no tiene fin y por esto nuestro pingüino no descansará hasta que se nos agote el tiempo durante la marcha o que hartos de jugar pulsemos el botón de reset.

Un dato curioso es que la versión europea del juego empieza por la etapa de 1500 kilómetros con final en el refugio de Estados Unidos, pero la primera etapa de la versión japonesa es Australia, es decir la número ocho, con un tramo de 1200 kilómetros.

Valoración personal

Konami creó una vez más un original y simpático juego que logra enganchar al jugador. La música es bastante repetitiva pero se puede aguantar bien (es un conocido tema). Los gráficos son bastante buenos con detalles como el movimiento de las nubes o la utilización de un pseudo-scaling para dar la sensación de aproximación a las cosas. Además es un juego rápido. Lástima que en este caso tampoco exista un final y que el jugador acabe aburrido de dar vueltas y vueltas.

Óscar Centelles



-USA- **Etapa 1** **Distancia 1500 Km.** **Tiempo 100 seg.**	**-UNITED KINGDOM-** **Etapa 6** **Distancia 500 Km.** **Tiempo 40 seg.**
-THE SOUTH POLE- **Etapa 2** **Distancia 1700 Km.** **Tiempo 120 seg.**	**-JAPAN-** **Etapa 7** **Distancia 2600 Km.** **Tiempo 165 seg.**
-USA- **Etapa 3** **Distancia 1100 Km.** **Tiempo 80 seg.**	**-AUSTRALIA-** **Etapa 8** **Distancia 1200 Km.** **Tiempo 90 seg.**
-USA- **Etapa 4** **Distancia 1200 Km.** **Tiempo 80 seg.**	**-AUSTRALIA-** **Etapa 9** **Distancia 1500 Km.** **Tiempo 100 seg.**
-ARGENTINA- **Etapa 5** **Distancia 1200 Km.** **Tiempo 80 seg.**	**-FRANCE-** **Etapa 10** **Distancia 1200 Km.** **Tiempo 90 seg.**

MadriSX 2004

El trayecto

Después de cenar con los hermanos de Eva en una pizzería de Sant Boi pudimos comprobar alucinados que uno de nuestros coches se lo había llevado la grúa mientras cenábamos. De todos modos no era el coche que íbamos a utilizar para el viaje, es decir, no era el mío; así que me partí la caja un rato, el dueño no se reía por eso, qué poco sentido del humor que tienen algunos... Tras las bromas y mofas de rigor, fuimos a buscarlo y el dueño del mismo recibió la correspondiente puñalada, después pudimos retirar el vehículo. Era hora de poner rumbo a Madrid, había quedado a las dos de la mañana con Jose y Raúl y empezaba a ser tarde. Eso pensaba yo hasta que llamo a Jose por teléfono y me dice que estaba durmiendo y que había puesto el despertador a las dos para hacer la mochila.

Pasados unos minutos de las dos, Eva y yo estábamos picando ante su puerta para que bajase; todavía no se que demonios llevaba Jose en aquella maleta enorme que pesaba un huevo cuando se suponía que íbamos a estar en el hotel únicamente una noche. Acto seguido fuimos a recoger a Raúl que venía con otros de tomarse unas copillas. En definitiva, que los cuatro partíamos hacia Madrid pasadas las dos y media.

El camino fue muy sencillo, solamente cogimos la autopista y para adelante. Hicimos dos paradas, la primera fue en Los Monegros; allí nos tomamos únicamente un café bien cargadito para mantenernos despiertos. Re-emprendimos nuestro camino hacia la capital y en en Zaragoza nos equivocamos y nos desviamos hacia Pontevedra. Es curioso que para volver hacia atrás tuvimos que tomar un cambio de sentido que ¡oh! casualmente se entraba tras un peaje. Pagamos, dimos la vuelta y volvimos a pasar por el peaje. Intentamos convencer al hombre que se encontraba en el puesto de cobro pero no sirvió de nada, se limitó a decirnos: hombre, ¡si hubierais preguntado! ¿Qué pasa, que tiene una trampilla secreta para hacer el cambio sin pasar por el peaje? Bueno, perdimos como una media hora larga por esta incidencia pero rápidamente volvimos a la carretera que nos interesaba.

La segunda parada la realizamos por Guadalajara. Allí cada uno pidió un menú especial desayuno por tres euros excepto uno de nosotros (y no soy yo) que pidió lo que después bautizamos como el menú gilipollas. Consiste en rechazar el menú de desayuno, pedir lo mismo que hay en el menú pero con una cosa menos y pagar 50 céntimos más de lo que pagan los demás, ¿a que está muy bien? Total, que nos volvimos a partir la caja a costa del pobre hombre.

MadriSX & Retro 2004

Pasados pocos minutos después de las 10 de la mañana llegamos al lugar del encuentro, que al igual que en estos últimos años se encontraba en el centro cultural El Greco. Pensábamos que el encuentro había comenzado ya y que llegábamos un poco tarde pero realmente la hora de apertura era a las 10:30 por lo que pillamos a Rafa y a otros acabando de montar las cuatro cosas que les faltaban. Saludamos a todo el mundo y comenzados a echar un vistazo por la sala. Lo primero que llamaba la atención, lógicamente, era que el MSX compartía habitación con otros sistemas, como por el Spectrum o Amstrad. Raúl y Jose, que era la primera vez que asistían a un encuentro de este tipo, alucinaban al ver los inventos y el partido que sacaban los usuarios a sus obsoletas máquinas.

Los stands

Empezando por el que se encontraba más cerca de la entrada pudimos encontrar:

Museo 8 bits: Stand presidido por Miguel Durán y su acompañante. Aquí podíamos ver diferentes consolas antiguas como la Atari Jaguar, Nintendo, Game Gear o la curiosa Sega Nomad. También había ordenadores como el Amstrad Plus, Dragón 64 o Apple II. Medio stand estaba ocupado por una legión de cartuchos de Master System y algunos otros sistemas como Megadrive o Saturn.
Miguel expuso, en la presentación al público de su stand, su actividad como coleccionista y vendedor de software y hardware. Además hizo un llamamiento a los usuarios que, como él, se dedican a coleccionar, para que se unan a la hora de



Mirando el stand de Ceinsur

hacer un pedido y así conseguir mejores precios. También propuso la idea de formar una organización de carácter no lucrativo con el fin de obligar a *Ebay* a la hora de hacer cumplir la leyes y obligar a los usuarios de este sistema a registrarse y dar a conocer su información personal, todo ello para evitar los fraudes. Un mínimo capital, simbólico, es necesario para registrar esta asociación de manera legal, comentaba Miguel.



Museo 8 bits

Podéis poneros en contacto con esta persona en los teléfonos 9155622918 ó 687466322. También podéis visitar su página en:

http://tienda.museo8bits.com

Vade-Retro & Older Machine: Otro stand dedicado a videoconsolas y ordenadores clásicos. Su propietario, Jorge exhibía un Commodore 16, una videoconsola Atari 2600 y una rara y vieja consola de Pong llamada Telstar entre otros aparatos. También tenían cintas y cartuchos de varios sistemas.

Speccy.org: Como ya habréis adivinado, mostraban varios Spectrum, entre ellos algunos grandes clásicos como el "teclas de goma". También habían otras cosas como una unidad de disco de 3'5" conectada a un Spectrum mediante un montaje casero. Además vendían multitud de juegos. Esta fue la primera vez que este grupo se presentó en este encuentro y esperan poder asistir muchas veces más. Daros un paseo por su completa página en:

http://www.speccy.org

Ceinsur: Presentaron varios Spectrums y Amstrad y tenían un gran surtido de cintas nuevas o como nuevas para vender a un precio realmente bajo. Miguel Ángel Montejo presentó también una cabina de máquina recreativa hecha a mano (utilizando madera). Dentro de ésta se escondía un PC con el emulador Mame funcionando. Después de varios intentos fallidos, esta versión de la cabina estaba lista para su venta. Si os interesa el tema visitad su página web:

http://www.ceinsur.net/ferias/msxr2004/

Karoshi Corporation: Stand dedicado a nuestro querido MSX, especialmente a la primera generación de la norma. Los chicos de Karoshi nos presentaron su disco compacto *Waver Game Pack*. Una recopilación de 79 juegos de MSX1 comprimidos utilizando la utilidad *WaveR 3* y grabados en un disco compacto como si uno de música se tratase. Utilizando un lector de compact disc cualquiera conectado al MSX a través de la entrada de datos del cassette podemos cargar los juegos a una velocidad de vértigo. Sus miembros explicaron al público que los juegos habían sido comprimidos hasta el punto que la carga de los mismos se realizaba a unos 5500 bps. También explicaron que habían conseguido cargas a velocidades cercanas a los 10Kbps pero que todavía estaban experimentando con ellas. El disco compacto con los juegos se podía comprar a un precio realmente barato, 3 euros, casi un regalo.

Una de las televisiones mostraba una demo de un juego en proyecto llamado "The Final Lap"; podíamos ver video en screen 3 con gráficos pre-renderizados que mostraba una secuencia de un duelo entre dos coches.

En su stand también podíamos coger gratuitamente un boletín informativo a todo color donde expresaban sus ideales y objetivos además de animar a la gente para que participe en la *MSXdev'04*, donde se pueden ganar interesantes premios. Su lema es ofrecer productos de calidad y gratuitos (freeware). Podéis visitar su página web en:



Stand de Karoshi Corporation

http://personal.telefonica.terra.es/web/karoshicorp/

Desgalitxat: Junto al anterior stand se encontraban nuestros ya habituales amigos. Como en cada encuentro presentaban multitud de camisetas, cuadros e ilustraciones con las portadas más espectaculares de juegos MSX así como la

conocida recopilación en varios discos compactos de un sinfín de portadas escaneadas en alta calidad, y todo a buen precio. También reservaron parte del stand para poner varios juegos a la venta, la mayoría japoneses. Además de estos juegos había un Turbo R GT, una buena oportunidad para hacerse con el MSX más deseado.

Finalmente en el último stand de **Power Replay** podíamos encontrar:

Martos trajo las revistas MSX Club, MSX Extra e Input MSX, todas ellas escaneadas, organizadas y presentadas en varios discos compactos que estaban a la venta. Una excelente manera de acceder y mantener para siempre a la prensa nacional sobre el MSX que se movía por aquella época.

En colaboración con el club Hnostar presentaron sin duda el producto estrella, *Bombaman*, del grupo Team Bomba. Lástima que no mostraron o dispusieron ningún MSX para mostrar las virtudes del juego pero el que había escuchado hablar de él no se lo pensó dos veces a la hora de comprar-


Stand de Desgalitxat

lo. Sin duda, una maravilla. Presentado en una caja de DVD con una excelente portada incluye dos diskettes con el juego, el manual y un disco compacto con multitud de extras como juegos *freeware* o la banda sonora de *Bombaman* grabada tal como suena por el Moonsound (que por cierto, es genial), entre otras cosas.

Destacar que cada miembro hizo una interesante presentación de su stand lo cual permitió conocer un poco más la idea y los objetivos de cada uno de ellos.

Se realizaron varios torneos; los juegos elegidos fueron el Moscow 2024, el Konami's Boxing y por último el Konami's Tennis (¡Martos perdió jugando a éste último!). Al mediodía se hizo una pausa para ir a comer.

De paseo por Madrid

A la hora de comer nos fuimos los cuatro al Kentucky que hay por la Gran Vía y después nos pegamos una buena siesta. El sonido de tambores que provenía de la calle nos despertó a eso de las ocho; lógicamente no regresamos a la reunión por la tarde pero es que necesitábamos dormir urgentemente tras el largo viaje. Más tarde, ya para cenar, quedamos con unos cuantos asistentes de la reunión y después de andar por medio Madrid acabamos donde empezamos cenando en un chino que hay cerca de la Gran Vía. Allí hablamos de muchas cosas y nos pusimos morados; Rafa se bebió dos chupitos a la vez, pero no eran de los fuertes (o sea, para nenas); vaya cara se le quedó.

Después del Chino nos quedamos los cuatro *"MSXeros"* y un *"Spectrumero"* que raptamos. Nuestro rehén Iñaki resultó ser un tío enrollado y nos tomamos unas *Murphy's* con él en una taberna irlandesa de por ahí. Estuvo muy bien. Después le dejamos en libertad e incluso nos pasamos las direcciones de e-mail.

Al día siguiente quedamos para dar unas vueltas por Madrid y visitamos el templo de *Debod*, los *jardines de Sabatini* y otros lugares emblemáticos. A la hora de comer acabamos en un Peruano donde comimos de maravilla y probamos la Crystal Cola (Inca Cola); alucinamos porque era amarilla y no estaba mal del todo. Incluso nos cantaron una Bachata (Fruta fresca, que la canta Carlos Vives). Vaya, que estuvo muy bien, os lo recomiendo.

Después de comer, y apresuradamente, nos despedimos y volvimos para Barcelona.

Mi valoración personal

Lo primero que pensé cuando supe que la reunión no iba a ser exclusivamente para el MSX es que quizás no tendría la gracia o espíritu de una reunión de las de siempre. En parte, sigo manteniendo esto, pero es lógico que se intenten buscar otras fórmulas para mantener con vida un evento como este. Es normal que con la poca asistencia de usuarios y clubs de MSX sea necesario contar con la ayuda de otros sistemas. De todos modos siempre me ha gustado ver todos los inventos y novedades de otros sistemas, así como conocer a la gente que se mueve alrededor de ellos (¡un saludo Iñaki!).

Como siempre me lo pasé fantásticamente. Por último agradecer a Rafa por buscarnos un sitio donde dormir a última hora y por el interés demostrado. Gracias también a todos los demás y enhorabuena por los 144 visitantes.

Oscar Centelles

Trucos

OUTRUN:

- “E” + “SPACE” : 4 segundos extra
- “F” + “K” + “SPACE” : Tiempo infinito (el contador sigue bajando, pero al llegar a 0 la partida no termina)
- “H” + “SPACE” : 1 segundo menos

Los trucos no son acumulativos.

Nota: hay que pulsar las teclas en la pantalla de presentación.

USAS:

No soy el único que ha pensado que este juego podría tener un final alternativo, al igual que otros juegos de Konami, ya que una explosión nuclear no parece un final feliz. Así que desensamblé parte del juego y descubrí que no había un final alternativo. Pero encontré que si te terminas el juego en un MSX japonés, se muestran unas pantallas que explican la explosión nuclear, y el “STAFF” del juego. (He adjuntado las dos pantallas que explican la explosión)

Está es la traducción de esos textos en japonés, hecha por Gunkan (Juan José Ferres):

“Después, cuando estaba investigando el material acerca de Usas que el profesor Atley había descubierto recientemente, descubrí que la estatua de Usas era el detonador de una bomba nuclear que una antigua civilización había desarrollado. El interruptor era la joya, y para evitar que se usase, se cuenta que la joya había sido dividida en cuatro piezas, repartidas por distintos lugares y selladas.”

Lo curioso o irónico de todo esto es que en la presentación del Usas aparece esto: “Si ellos encuentran las 4 piezas de la piedra preciosa, algo bueno ocurrirá” No sé si es que tenían pensado otro final para el juego, o que tienen un sentido del humor un poco extraño.

PENGUIN ADVENTURE:

El mayor misterio que ha acompañado a este juego ha sido como ver el final bueno, en el que llegamos a tiempo para salvar a la princesa. Muchas han sido las teorías: coger todos los atajos, no jugar a las tragaperras, no pausar el juego, etc... Pero la única y verdadera razón para poder conseguirlo es esta:

Pausar el juego (4 x n) + 1 veces. Por ejemplo:
(4 x 0) + 1 = 1 vez
(4 x 1) + 1 = 5 veces
(4 x 2) + 1 = 9 veces
(4 x 3) + 1 = 13 veces
Tenéis que tener en cuenta que si continuáis la partida el contador se reinicia a 0.

NEMESIS 3:

Son de sobra conocidos los “passwords” de este juego, pero no lo es tanto lo que hay que hacer para que nos los digan. Los requisitos son, tener el Game Master 2 en el otro slot, y habernos terminado el juego. Al comenzar una nueva partida nos aparecerá una pantalla con los “passwords”

FIND: Nos muestra donde hay armas o mapas escondidos. Similar al “extra sensor”
EXPAND: La barrera protectora (forcefield) dura el doble
HARD: Modo de juego difícil
GOOD: Modo de juego fácil

Manuel Pazos

Screen 2 a fondo

Inauguramos en este número una sección que no pudo aparecer en el primero por falta de tiempo: OPCODE. En esta sección se intentará en cada número ofrecer algún aspecto relacionado con la programación del sistema MSX, normalmente en ensamblador, aunque no se descartan posibles artículos que traten con otro lenguaje como pudiera ser Basic, C o incluso Pascal.

Tengo que advertir al lector que el autor del artículo, yo mismo, no soy ningún entendido en lo que se refiere a todos los aspectos de programación del MSX, si no más bien quiero poner en conocimiento de los demás aquél que yo mismo he adquirido mediante la lectura, la búsqueda y la consulta a otras personas. Es por ello, y por pertenecer a la misma condición humana que los demás, que en presente artículo podrán existir errores, e incluso muchos de vosotros encontrareis alternativas más optimas a las que aquí se os ofrece. Por ello quiero hacer diversos comentarios: el primero, que no me puedo hacer responsable d e las fatalidades que un error pueda provocar en vuestro sistema, auque sí tengo que decir que los programas aquí reproducidos han sido probados con éxito en mi propio MSX; y segundo, que no prometo que vayáis a aprender la mejor forma de cada uno de los aspectos de programación que en éste y los siguientes artículos se traten. Mi idea es simplemente ayudar a aquellos que tienen cierta dificultad programando el sistema y que se han encontrado con dudas y preguntas cuyas respuestas no han sido todavía halladas. Por otra parte, cualquier error encontrado, sugerencia e incluso crítica (abstenerse descalificaciones) podrán ser dirigidas a yakumoklesk@yahoo.es.

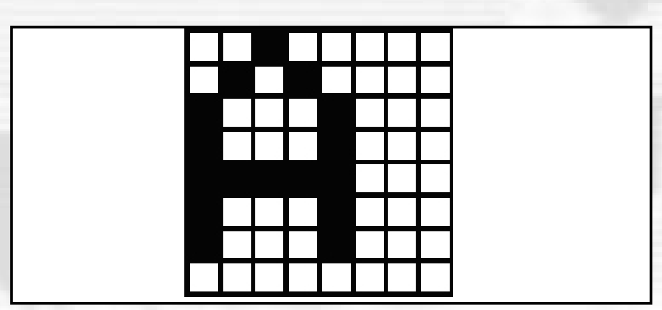
Durante el transcurso del corto, pero intenso tiempo que ha pasado desde que me reenganché al sistema hasta la fecha actual, muchas consultas y dudas se han expuesto en foros, canales y grupos de correos acerca de aspectos de programación del sistema MSX. Uno de ellos es precisamente el que trata con los aspectos del VDP y modos de pantalla en general, y más en concreto el SCREEN 2, por ser uno de los mejores modos para los MSX de primera generación, y por tanto compatible en cualquier otro modelo.

En este artículo se pretende tratar y explicar el funcionamiento de este modo de vídeo, rutinas para el movimiento y copia de datos a la memoria de vídeo o VRAM y algunos aspectos del VDP que permitirán al programador principiante tener un suficiente dominio del screen 2 como para poder realizar juegos simples que no requieran scroll. De todas maneras, la forma en que se va a explicar el funcionamiento del screen 2 será mediante la realización del código para un editor de patrones en este mismo modo de vídeo. El usuario podrá editar sus propios patrones y guardarlos en un archivo a disco para que después lo pueda usar para sus propias aplicaciones gráficas o juegos.

Sobre la sintaxis: el código fuente proporcionado es compatible con la versión 3 del Chaos Assembler, un ensamblador cruzado para sistemas PC con sistema operativo Windows que usa el ensamblador tasm para generar código z80. Éste y no otro es el tipo de sintaxis que voy a usar pues es el que utilizo habitualmente para programar. El por qué no utilizo un ensamblador para MSX radica en la lentitud de éstos para generar código objeto cada vez que se procede a ensamblar. Gracias a la velocidad del PC puedo hacer las pruebas de manera mucho más rápida y dedicar más tiempo a lo verdaderamente importante: programar. Tengo que advertir, pues, que no voy a cambiar el tipo de código presentado en éste ni en próximos artículos, y cada uno deberá de realizar por sí mismo las modificaciones pertinentes para adaptarlo al entorno en que trabaje.

Y ahora sí, adentrémonos en lo que hace unos cuantos párrafos estabais esperando. Veamos primeramente algunas características del screen 2.

-El screen 2 se corresponde con el modo gráfico del VDP número 2
-La pantalla consta de 32 patrones horizontales por 24 verticales. Se pueden utilizar 16 colores simultáneamente de una paleta de 512, salvo en sistemas MSX de primera generación, en los que la paleta es fija y no puede ser modificada.
-Patrones. Los patrones equivalen a las unidades de información del modo de vídeo. El símil más práctico con el que se puede comparar un patrón equivaldría a la tabla de bits que define un carácter como por ejemplo el de la letra A (ver ejemplo de abajo). Los patrones de screen 2 tienen un tamaño de 8 por 8 puntos, y por cada 8 puntos horizontales de un patrón, 2 colores distintos pueden usarse simultáneamente.



Patrón para el carácter A

-Requerimientos de memoria: 6144 bytes para la definición de los patrones, 6144 bytes para la definición de los colores de los patrones
-Modo de sprite 1 (se tratará en este artículo de forma básica)

Uno de los aspectos que puede resultar de más difícil compresión es la del concepto de patrón. Habrá algunos que habiendo leído las líneas anteriores ya se haya hecho una idea de lo que quiere decir, o ha entendido el mecanismo por el que vemos caracteres en pantalla, pero voy a intentar en las siguientes líneas ser algo más preciso al respecto.

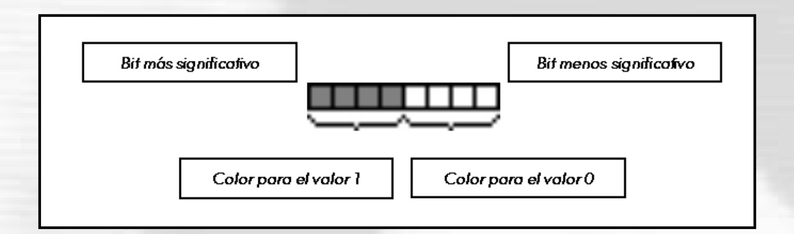
Para mostrar una letra o un número por pantalla y que el usuario la pueda leer, es necesario poder decirle al VDP que active unos puntos de la pantalla (los que conforman el dibujo de la letra) mientras que otros puntos no sean activados. Siendo un poco más precisos, debemos decirle concretamente que dibuje los puntos que forman parte del dibujo de la letra o número de un color, y que dibuje los puntos que no forman parte de dicha letra o número de otro color distinto. En el ejemplo anterior, los puntos que forman el dibujo de la letra A (de aquí en adelante adoptaré el

término carácter para referirme tanto a letras, como a números, como a signos de puntuación, etc.) se pintan de color negro, mientras que los puntos de fondo se pintan de color blanco. Aunque en realidad veáis una rejilla, ésta ha sido pintada para intentar clarificar el diseño del carácter. Así pues, necesitamos decirle al VDP esa información, es decir, para cada uno de los caracteres, debemos especificar una tabla que indique (siguiendo el ejemplo anterior) qué punto debe ser pintado de negro, y qué punto debe ser pintado de blanco. Tomemos por ejemplo que el punto negro se codifica con un 1, y el punto blanco se codifica con un 0. Cada una de las 8 líneas del carácter en cuestión necesitarán exactamente un byte para poder ser definida. Así, la primera línea equivaldría al byte 00100000 en binario, o 20h en hexadecimal. Es evidente pues, que para cada carácter, serán necesario 8 bytes, pues cada carácter consta de 8 líneas que necesitan de 1 byte para ser definidas.

Toda esta información se debe de almacenar para cada uno de los caracteres que queramos definir. El inicio del lugar donde deben de ser almacenada esta información se haya en la dirección de memoria de vídeo 0000h (ojo, en VRAM, no en RAM). Pero con especificar estos datos no basta. Hay que decirle al VDP, una vez definido el patrón de un carácter, dónde debe mostrarlo. Para ello, se le debe de especificar al VDP que el número de patrón que hemos definido debe de pintarlo en una posición que nosotros le especifiquemos. Ello se hace copiando el byte correspondiente al número de patrón definido en la zona de memoria de vídeo comprendida entre 1800h y 1AFFh. Si restáis 1AFFh – 1800h veis que el resultado es exactamente 768, es decir, 32x24, que son todos los caracteres que caben en una pantalla de screen 2. Así pues, si queremos decirle que en la fila 10, columna 12 se muestre nuestro carácter (la A, cuyo código ASCII es 65), deberemos almacenar un 65 en la posición de memoria de vídeo correspondiente a esa fila y columna. Puesto que cada fila consta de 32 caracteres, si consideramos la matriz como un vector lineal, que es como realmente está organizado en la memoria de vídeo, la columna 12 de la fila 10 se sitúa en la posición 332 (10x32 + 12) a partir de la dirección 1800h, es decir, se encuentra en 1800h + 332 (14Ch) = 194Ch. Es en esta posición de la VRAM donde debemos de almacenar el byte de valor 65.

Además de ello, tenemos que tener en cuenta que el carácter 65 debe de estar definido en la correspondiente zona de la VRAM que corresponde a la generación de los patrones. Puesto que cada carácter se define con 8 bytes, 8x65 = 520 es la posición a partir de 0000h donde se tendrán que definir los 8 bytes correspondientes al carácter 65 (nuestra querida A). Lamentablemente esto aún no es suficiente para que podamos ver un carácter en pantalla. Hay que especificar también el color que tiene este carácter. Y es que nosotros, al especificar los 8 bytes del carácter, hemos escrito los 0's y los 1's pensando alegremente que el 0 se pintaría de blanco y el 1 de negro. Pero lo cierto es que al iniciar el modo de vídeo, no existe información sobre los colores que toma el valor 0 o el 1, y se los tenemos que especificar nosotros. La tabla de colores empieza en 2000h. Para cada carácter, se tiene que especificar 2 colores por cada línea del carácter, que serán el color con el que se pinte el punto que toma valor 1, y el color con el que se pinte el punto que toma valor 0. Para especificar este color, hay que decirle el índice de la paleta de donde se tomará la información. Puesto que la paleta es de 16 colores, necesitamos 4 bits para especificar 16 valores diferentes. Y puesto que se tienen que especificar 2 colores diferentes por línea de carácter, necesitaremos 2x4bits = 1 byte para guardar la información de color de cada línea de un carácter. Es decir, necesitaremos 8 bytes para almacenar toda la información de color de un carácter.

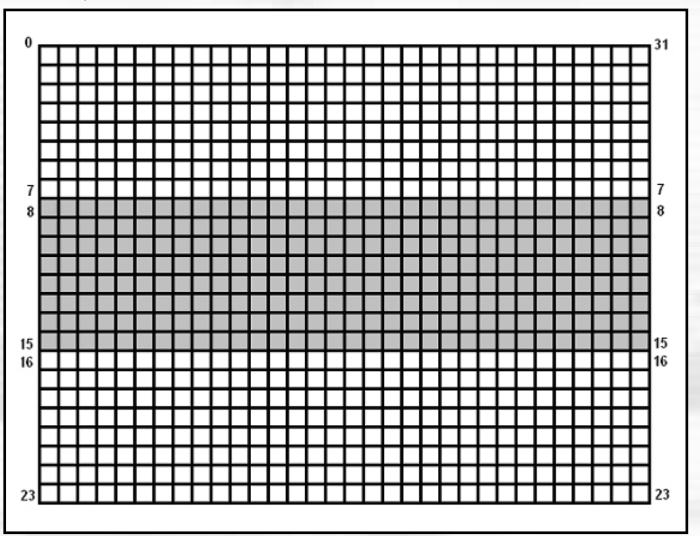
El byte con información de color, se organiza de la siguiente manera:



Calculemos la dirección de la VRAM donde tendremos que almacenar los 8 bytes con información de color. Habíamos calculado que 65x8 era 520. Puesto que la tabla de colores empieza en 2000h, 2000h + 520 (208h) = 2208h.

Una vez especificado los 8 bytes de la definición del patrón del carácter, los 8 bytes de la definición de los colores del carácter, y el byte correspondiente a la localización en la pantalla del carácter, podremos disfrutar de una deliciosa A flotando en nuestra pantalla.

Antes de ponernos a hablar del VDP y de cómo escribir en memoria de vídeo, quiero explicar algunas cosas más que quizá algún lector haya notado. Si para especificar una posición de un carácter, tenemos que escribir 1 byte, sólo podremos especificar posiciones de 256 caracteres diferente, y sin embargo en las especificaciones he dicho que se pueden especificar hasta 768 (3 veces 256).
Bien. En realidad, lo que ocurre es que no existe una tabla de patrones, sino 3. Y no existe una tabla de colores, sino 3. Y no existe una zona de pantalla donde especificar la posición del carácter, sino 3.



Subdivisión de la pantalla de vídeo de screen 2. Cada zona dispone de 256 caracteres diferentes e interdependientes entre zonas.

Cada zona dispone de su propia tabla de patrones así como de su propia tabla de colores, completamente interdependientes entre ellas. Es decir, que en la primera tabla podemos especificar patrones que no aparezcan en la segunda tabla o en la tercera, y podemos especificar colores para los patrones diferentes de los definidos en el resto de tablas. Ello implica que podamos definir 768 caracteres diferentes, como ya se dijo al principio, pero conlleva una contrariedad. Los caracteres de una tabla concreta, sólo pueden aparecer en la zona específica de la pantalla. Es decir, los patrones definidos en la primera tabla, sólo podrán mostrarse en el primer tercio de la pantalla, los de la segunda tabla, en el segundo tercio, y lo mismo para la tercera tabla. Así pues, si queremos mostrar un mismo carácter en las tres zonas de pantalla simultáneamente, ha de ser definido también en cada una de ellas. Este sistema puede considerarse un poco engorroso, pero también puede tener ciertas ventajas. Por ejemplo, podemos realizar un juego en el que la parte primera y segunda se use para gráficos, y la última parte se use para texto, como si de una aventura conversacional se tratara. Esto implicaría que únicamente deberíamos de definir texto en la tercera tabla de patrones y no en las otras dos, que la dejaríamos para definir patrones gráficos.

Existe de todas maneras, un truco que aquí no voy a explicar, consistente en aprovechar un fallo del VDP que hace que únicamente definiendo la primera tabla de caracteres, estos puedan ser usados en el resto sin tenerlas que definir de nuevo, pero esto limita el máximo de caracteres definidos a 256. Veamos pues ahora cómo quedan definidas las zonas de memoria de vídeo para la especificación de la posición de los caracteres, la definición de los patrones y la especificación de los colores para cada uno de éstos en sus tres respectivas zonas. La zona donde se especifica la posición donde se debe de mostrar un carácter recibe el nombre de tabla de nombres de patrón, porque en realidad, el número entre 0 y 255 que especificamos no es más que el nombre que recibe el patrón.

Tabla de patrones 1: Inicio: 0000h Final: 07FFh
Tabla de patrones 2: Inicio: 0800h Final: 0FFFh
Tabla de patrones 3: Inicio: 1000h Final: 1FFFh
Tabla de nombres de patrón 1 (zona 1 de la pantalla): Inicio: 1800h Final: 18FFh
Tabla de nombres de patrón 2 (zona 2 de la pantalla): Inicio: 1900h Final: 18FFh
Tabla de nombres de patrón 3 (zona 3 de la pantalla): Inicio: 1A00h Final: 1AFFh
Tabla de colores 1: Inicio: 2000h Final: 27FFh
Tabla de colores 2: Inicio: 2800h Final: 2FFFh
Tabla de colores 3: Inicio: 3000h Final: 37FFh

Otras direcciones interesantes son las siguientes:

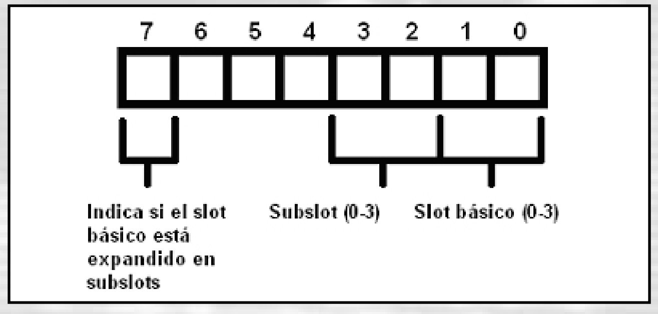
Tabla de atributos de sprite: Inicio: 1B00h Final: 1B7Fh
Tabla de definición (generadora) de sprites: Inicio: 3800h Final: 3FFFh
Tabla de paleta (en MSX de primera generación no se puede redefinir): Inicio: 1B80h Final: 1BAFh

Bien, tras este repaso interesante sobre la disposición de la información en la memoria de vídeo cuando screen 2 está activado, vamos a ver cómo hacer que nuestro MSX pueda poner el modo de vídeo y definir los patrones. Algunos de los procesos se realizarán usando llamadas a la bios, otros se realizarán mediante el acceso directo a los registros del VDP usando evidentemente el lenguaje estrella: ensamblador.

Para cambiar el modo de vídeo, usamos una función que nos proporciona la BIOS del MSX, llamada CHMOD, cuya dirección dentro de la zona de memoria donde se encuentra la BIOS es 005Fh. De todas maneras, antes de continuar, quiero decir que el editor que he realizado funciona bajo MSX-DOS, con lo cual hay que tener en cuenta que para realizar llamadas a la BIOS hemos de utilizar la función de llamadas a los slots (interslot-call) CALSLT, cuya dirección de memoria estando en MSX-DOS es 001Ch. La función interslot-call funciona de la siguiente manera:

Registro IX: debe tener la dirección de la función que se encuentra en otro slot y que queremos llamar
Registro IY: debe contener en la parte alta, la información sobre el slot y subslot donde se encuentra dicha función
Resto de registros: debe contener la información necesaria y requerida por la función del otro slot a la que queremos llamar

En nuestro caso, IX deberá tomar por valor CHMOD, es decir, 005Fh. La función CHMOD requiere que en A se especifique el modo de vídeo al que queremos cambiar, en nuestro caso, A valdrá 2, puesto que queremos activar screen 2. El registro IY tiene que tener en su parte alta la información sobre el slot y el subslot donde se encuentra CHMOD, es decir, el slot y subslot donde se encuentra la BIOS. En los MSX, esta información se halla en EXPTBL, localizada en la zona de trabajo (RAM) FFCCh, y es un byte que tiene la siguiente información:



Los dos primeros bits, indican el slot donde se encuentra la BIOS. Los bits 2 y 3, a su vez, indican el subslot en el que se encuentra si es que el slot básico está expandido. Esta información, sobre si el slot básico está expandido o no, se encuentra en el bit 7 del mismo byte. Veamos pues la secuencia de instrucciones para poner el modo de vídeo screen 2.

```
ld      A, 2            ; Modo de vídeo screen 2
ld      IX, CHMOD       ; Rutina de la BIOS para el cambio de modo de vídeo
ld      IY, (EXPTBL-1)  ; Hay que colocar el byte en la parte alta (de ahí el –1)
call    CALSLT          ; Llamada interslot a CHMOD
```

Para poner el modo de vídeo screen 0, simplemente hay que cambiar el 2 por un 0, y saldremos a modo texto. A continuación veremos aspectos básicos del VDP para poder tratar los temas de

la memoria de vídeo. Básicamente trataremos los aspectos que se refieren a la escritura y lectura de memoria de vídeo.

El VDP dispone de distintos registros que se organizan para realizar tareas muy distintas. Algunos de ellos se usan para un único cometido, otros para varios, y a veces ocurre que diversos registros se deben de configurar para realizar una tarea común. Existen registros de estados, cuya información se usa para saber qué está ocurriendo en cada momento en el procesador de vídeo, y otros de ellos se utilizan como registros de escritura, denominados registros de control, para realizar diversas configuraciones, especificaciones de direcciones de memoria de vídeo y otras tareas diversas. Los registros que nos incumben a nosotros son los que tienen que ver con direcciones de memoria de vídeo, además de la escritura y lectura en/de éstas.

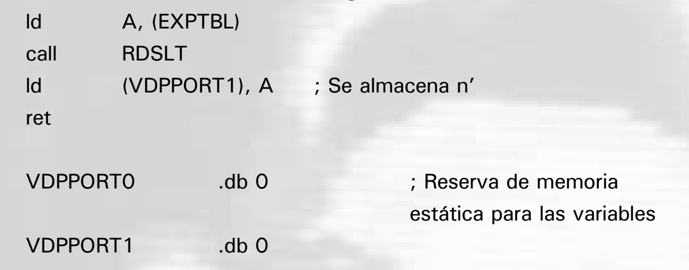
Para leer el contenido de un registro de estado, o bien acceder a un registro de control, es necesario trabajar con unos puertos de que dispone el VDP. Dependiendo de qué tipo de operación queramos realizar, y a qué registro del VDP queramos acceder, deberemos usar unos u otros puertos. Los puertos del VDP son básicamente cuatro, y dependiendo de si van a ser usados para leer de ellos o para escribir en ellos, las direcciones de estos puertos pueden ser diferentes.

Puerto 0: Lectura	n	Lectura de VRAM
Puerto 0: Escritura	n'	Escritura a VRAM
Puerto 1: Lectura	n + 1	Lectura del registro de estado
Puerto 1: Escritura	n' + 1	Escritura a un registro de control
Puerto 2: Escritura	n' + 2	Escritura al registro de paleta
Puerto 3: Escritura	n' + 3	Escritura indirecta a un registro

Los valores de n y n' normalmente son siempre los mismos, el número 98h, pero existen algunos MSX, aunque pocos, cuyas direcciones son diferentes. Es por esto que, para compatibilidad con todos los MSX, se debe de usar el valor que hay almacenado en cierta dirección de la ROM principal del MSX. El valor n se halla en el byte 6 de la MAIN-ROM, mientras que el valor para n' se halla en el byte 7. El cómo leer estos valores es muy sencillo. De la misma forma que antes hemos realizado una llamada a una función que está en otro slot, se pueden leer valores o escribir éstos en zonas de memoria que se hallen en otro slot. Para ello se usan las llamadas RDSLT, situada en 000Ch y WRSTL, que se encuentra en 0014h. Nosotros sólo usaremos en este caso la primera función para obtener los valores n y n' de la ROM principal. Veamos el código:

```
GetVDPPorts:

ld      HL,     6           ; Dirección de n (dirección
                              a leer)
ld      A, (EXPTBL)         ; A debe contener el slot
                              de donde leer (MAIN-ROM)
call    RDSLT               ; Lectura
ld      (VDPPORT0), A       ; El resultado, obtenido en
                              A, se guarda
```



```
inc     HL                  ; Siguiente dirección (7)
ld      A, (EXPTBL)
call    RDSLT
ld      (VDPPORT1), A       ; Se almacena n'
ret

VDPPORT0    .db 0           ; Reserva de memoria
                              estática para las variables
VDPPORT1    .db 0
```

Cada vez que realicemos un programa que vaya a usar funciones que usen los puertos del VDP, deberemos de llamar a esta rutina para inicializar los valores correspondientes a las direcciones bases de estos puertos, de lo contrario no obtendremos ningún resultado. La nomenclatura aquí usada quizá pueda confundir al lector, pues se ha usado PORT0 y PORT1 para hacer referencia a n y n'. Sea como fuere, se ha de tener en cuenta para futuros usos de estos valores.

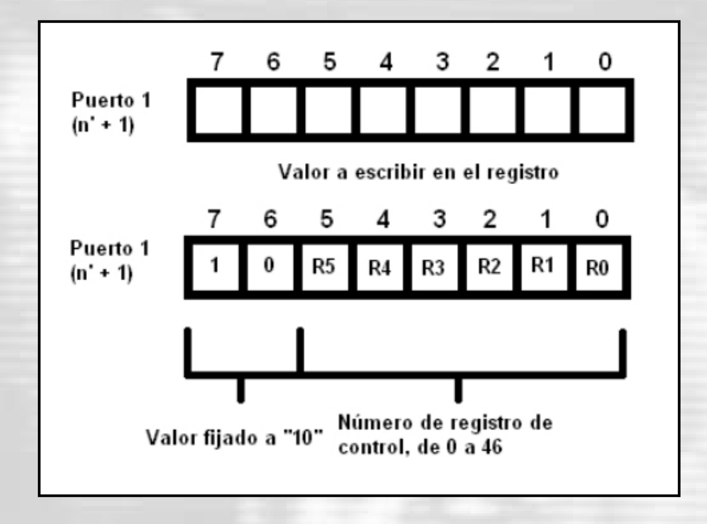
Ahora veremos como, usando estos valores previamente almacenados, se puede conseguir escribir y leer valores de la memoria de vídeo, pues es una de las operaciones que más a menudo realizaremos. Para ello antes hay que explicar cómo deben usarse estos puertos para escribir en los registros de control los valores de las direcciones, pero antes, hay que saber cómo especificar no sólo el valor que queremos escribir, sino el registro de control con el que queremos tratar.

Acceso a los registros de control.

A lo largo de este tutorial, veremos la necesidad que tenemos de acceder a estos registros para realizar operaciones con la memoria de vídeo. Existen diferentes modos de acceso a estos registros, que veremos a continuación:

Acceso directo:

En este modo de acceso, primeramente se tiene que especificar el dato que se quiere escribir en un registro de control, para inmediatamente enviar el número de dicho registro de forma que el valor se almacene en él. Ambos datos se escriben en el puerto 1 (n' + 1).



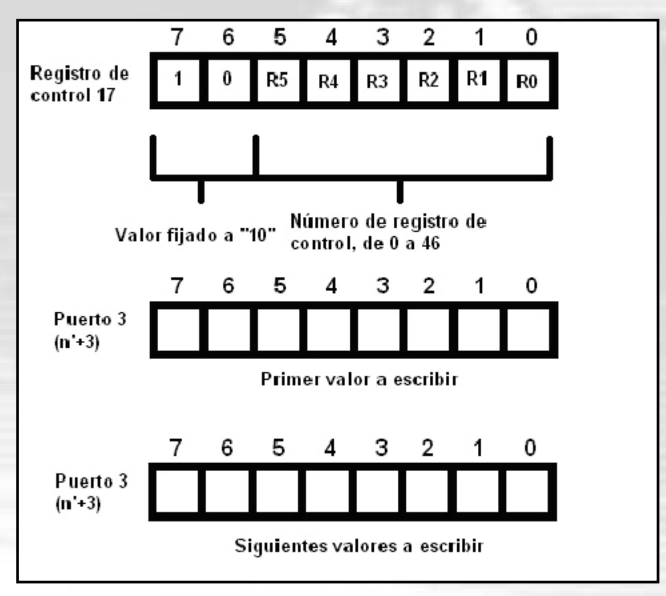
Veamos un ejemplo de cómo escribir en el registro de control 10 el valor 30.

```
ld      A, (VDPREG1)    ; A tiene n'
inc     A               ; A = n' + 1, dirección de escritu
                          ra del Puerto 1
ld      C, A            ; C se usa para hacer out's a los
                          puertos
ld      A, 30           ; Valor a escribir en el registro de
                          control 10
out     (C), A          ; Se envía el valor al Puerto 1
ld      A, 10 + 80h     ; Registro de control al que quere
                          mos enviar el valor
                        ; el 80h hace que el bit 7 tenga el
valor 1
out     (C), A          ; Escritura en el Puerto 1.El registro
                         10 toma el valor 30
```

Parecen muchas instrucciones para una simple escritura, pero una vez os acostumbráis sale prácticamente solo puesto que no es tan difícil, ¿verdad?

Acceso indirecto (modo de no autoincremento):

Para usar este modo de escritura, primero tenemos que escribir en el registro de control 17, el número de registro control al que queremos enviar los datos, teniendo en cuenta que el bit 7 debe de tener el valor 1, y el bit 6 el valor 0. Una vez el registro 17 tenga el valor del registro de control deseado, se usa el puerto 3 para escribir los datos sucesivamente. Este modo se usa para escribir más de un dato seguido al mismo puerto, pues a veces resulta ser necesario.



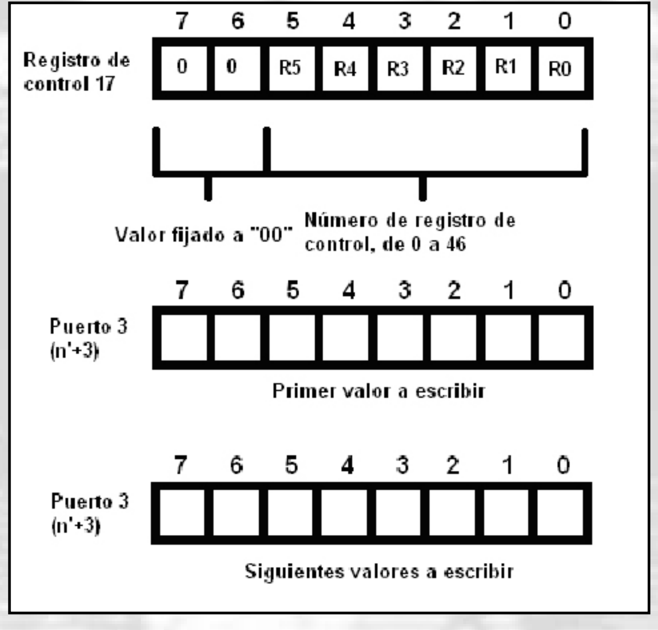
Veamos un ejemplo de cómo enviar los valores 3, 7 y 10 al registro de control 12.

```
ld      A, (VDPREG1)    ; A = n'
inc     A               ; A = n' + 1 (Puerto 1, escritura)
ld      A, C
ld      A, 12 + 80h     ; El número de registro de control
                          que queremos escribir en
                          el registro de control 17
out     (C), A
ld      A, 17 + 80h     ; El registro 17 tendrá el valor 12 +
80h
inc     C               ; C = n' + 2 (Puerto 2, escritura)
inc     C               ; C = n' + 3 (Puerto 3, escritura)
ld      A, 3            ; Primer valor
out     (C), A
ld      A, 7            ; Segundo valor
out     (C), A
ld      A, 10           ; Tercer valor
out     (C), A
```

La inicialización es un poco engorrosa al principio, pero luego podemos escribir una serie de valores con una facilidad muy grande, pues simplemente debemos inicializar el registro A con el valor y enviarlo directamente.

Acceso indirecto (modo de autoincremento):

Pocas variaciones en cuanto al modo de escribir los datos a los puertos. También se usa el registro de control 17 para especificar de forma indirecta el número de registro de control al que queremos enviar datos. Esta vez, el bit 7 y 6 deben tomar ambos el valor 0. Otra diferencia esencial es que cada vez que se envíe un dato al puerto 3, el número de registro almacenado en el registro de control 17 se verá automáticamente incrementado en una unidad.
A continuación os muestro un ejemplo de cómo escribir el valor 3



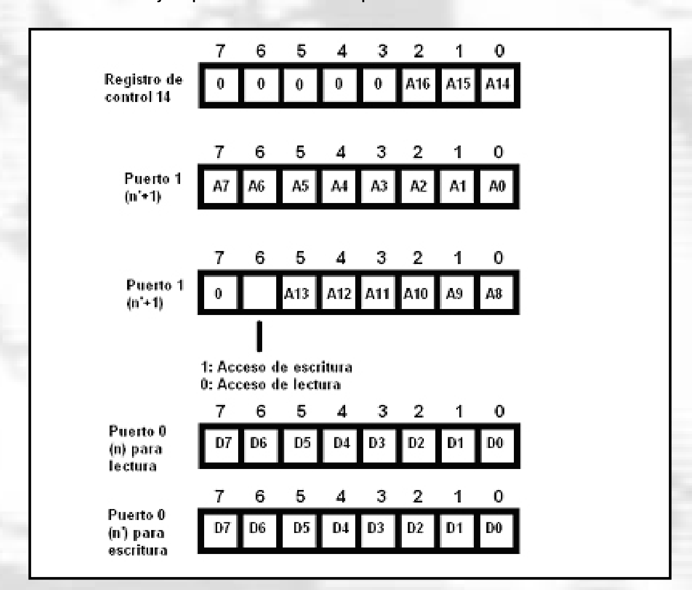
en el registro 33, el 5 en el registro 34, el 9 en el registro 35 y el 11 en el registro 36.

```
ld      A, (VDPREG1)    ; A = n'
inc     A               ; A = n' + 1 (Puerto 1, escritura)
ld      A, C
ld      A, 33           ; El número de registro de control
                          que queremos escribir en
                        ; el registro de control 17 (esta vez
```

```
                                 sin el 80h)
out         (C), A
ld          A, 17 + 80h          ; El registro 17 tendrá el valor 12 +
                                 80h
inc         C                    ; C = n' + 2 (Puerto 2, escritura)
inc         C                    ; C = n' + 3 (Puerto 3, escritura)
ld          A, 3                 ; Primer valor en 33
out         (C), A               ; El valor del registro 17 es ahora
                                  34
ld          A, 5                 ; Segundo valor
out         (C), A               ; El valor del registro 17 es ahora
                                  35
ld          A, 9                 ; Tercer valor
out         (C), A               ; El valor del registro 17 es ahora
                                  36
ld          A, 11                ; Cuarto valor
out         (C), A               ; El valor del registro 17 es ahora
                                  37
```

A pesar de que he explicado estos tres modos de direccionamiento, normalmente usaremos el primero, pero he querido explicarlos aquí por si en futuros tutoriales tratara otros temas del VDP, como el modo gráfico de los msx2.

Una vez familiarizados con los modos de escritura y con la forma de enviar datos a los diferentes puertos mediante los valores calculados al inicio, veamos cómo escribir datos en la memoria de vídeo. Para llevar esta operación a cabo, en el registro de control 14 se debe de especificar la página de VRAM sobre la que queremos escribir o bien leer un dato. Este número de página corresponde a los 3 bits más significativos de la dirección de memoria de vídeo. Puesto que muchos MSX disponen de 128ks de VRAM, se necesitan 17 bits para especificar una dirección si pensamos en que la memoria de vídeo se distribuye linealmente. Así pues, los bits A16 a A14 definen la página de VRAM.
Una vez especificada la página (en este tutorial trabajaremos siempre con la 0), debemos de escribir los 8 bits menos significativos (A7 a A0) en el Puerto 1, para después enviar el resto de bits (A13 a A8) otra vez de nuevo al Puerto 1. Pero en el mismo momento de enviar estos últimos bits, deberemos hacer que el bit 6 del Puerto 1 tome el valor 1 si pensamos escribir en la VRAM, o 0 si deseamos leer de ella.
Veamos un ejemplo de una rutina para escribir un valor en la



memoria de vídeo, rutina que podremos guardar para usarla en futuros accesos.

```
;———————————————————————-
; Nombre:
;           onwrvrm: (own nwrvrm) Propia nwrvrm
; Descripción:
;           Igual que la de la BIOS
; Entrada:
;           HL: dirección de VRAM a la que escribir
;           A: Dato a escribir
; Salida:
;
; Registros afectados:
;           F
;———————————————————————-
onwrvrm:
            push    BC          ; Salvamos BC porque lo usamos
            ld      B, A        ; Salvamos el dato en B
            ld      A, (VDPPORT1)     ; A = n'
            ld      C, A
            inc     C           ; C = n' + 1 (Puerto 1, escritura)
            ld      A, H
            rlc     A
            rlc     A
            and     03h         ; A = página
            di                  ; Deshabilitamos interrupciones
            out     (C), A                ; escritura de la página
            ld      A, 80h + 14           ; Registro 14, acceso
                                           directo
            out     (C), A                ; Escritura del valor
            ld      A, L                  ; 8 bits menos significa-
tivos de la dirección
            out     (C), A                ; Escritura en el Puerto 1
(ya seleccionado en C)
            ld      A, H                  ;
            and     03fh                  ; Bits A13 a A8
            or      40h                   ; Bit 6 a 1 indica que
queremos escribir
            out     (C), A
            ei                  ; Rehabilitamos interrupciones
            ld      A, (VDPPORT1)     ; A = n' (Puerto 0, escrit-
ura)
            ld      C, A
            ld      A, B                  ; A toma el valor a
escribir
            out     (C), A                ; A -> VRAM
            pop     BC                    ; Restauramos los
registros afectados
            ret                           ; Volvemos de la llamada
```

Esta función se puede copiar tal cual para convertirla en una que lea un valor de la VRAM. Para ello, la instrucción *or 40h* necesita ser eliminada, y el último *out (C), A* debe ser sustituido por un *in A, (C)*. Hay veces que necesitaremos escribir un mismo byte diversas veces en sucesivas posiciones de memoria. Para ello se puede usar la función anterior en un bucle, o bien hacer una función nueva que aproveche el autoincremento automático de la dirección de la memoria de vídeo. Veámoslo:

```
;—————————————————————-
; Nombre:
;	stosbvrm (stores string of bytes) Almacena bytes en VRAM
; Description:
;	Copia el contenido de A en posiciones sucesivas de la VRAM
; Entrada:
;	HL: Dirección destino de VRAM
;	A: Byte a escribir
;	BC: Contador con el número de veces que debe escribirse el byte
; Output:
;
; Registers affected:
;	AF
;	BC
;	HL
;—————————————————————-
stosbvrm:
	push	AF		; Saves value
	push	BC		; Save counter
	ld	A, (VDPPORT1)	; A = n'
	ld	C, A
	inc	C		; C = n' + 1 (Puerto 1, escritura)
	ld	A, H
	rlc	A
	rlc	A
	and	03h		; A = página
	di			; Deshabilitamos interrupciones
	out	(C), A		; escritura de la página
	ld	A, 80h + 14	; Registro 14, acceso directo
	out	(C), A		; Escritura del valor
	ld	A, L		; 8 bits menos significativos de la dirección
	out	(C), A		; Escritura en el Puerto 1 (ya seleccionado en C)
	ld	A, H		;
	and	03fh		; Bits A13 a A8
	or	40h		; Bit 6 a 1 indica que queremos escribir
	out	(C), A
	ei			; Rehabilitamos interrupciones
	ld	A, (VDPPORT1)	; A = n' (Puerto 0, escritura)
	ld	C, A
	pop	HL		; Restaura el contador en HL
	pop	AF
	ld	B, A		; Salva el valor en B
stosbvrm_outloop:
	ld	A, B		; Recupera el valor
	out	(C), A		; A -> VRAM
	dec	HL
	ld	A, L
	or	H		; ¿Hemos acabado?
	jr	NZ, stosbvrm_outloop	; Si no, damos otra vuelta
	ret			; Retornamos del procedimiento
```

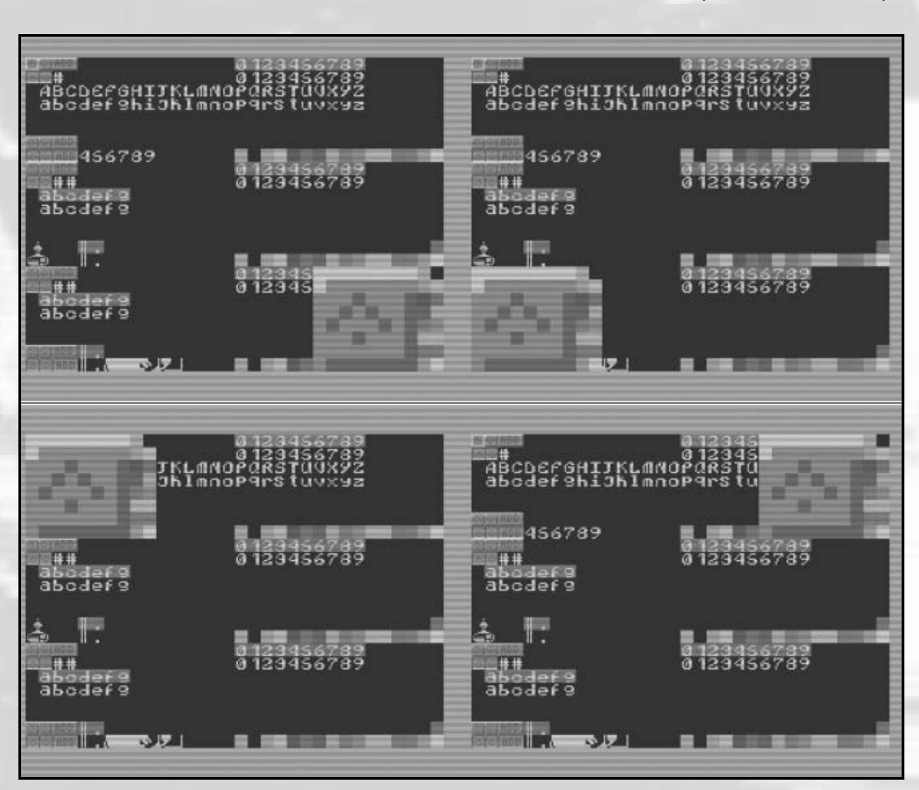
Podemos aprovechar esta función para almacenar valores que se vayan autoincrementando automáticamente, simplemente introduciendo la instrucción *inc B* entre el par de instrucciones *ld A, B* y *out (C), A* justo tras la etiqueta del bucle. Otras funciones que suelen ser necesarias son las de movimientos de zonas de memoria de vídeo a memoria principal y viceversa. Veremos en el siguiente ejemplo una función para leer una serie de bytes de VRAM y almacenarla en una zona de la memoria.

```
;—————————————————————-
; Nombre:
;	oldirmv (own ldirmv) Propia ldirmv
; Descripción:
;	Igual que la BIOS
; Entrada:
;	HL: Dirección VRAM origen
;	DE: Dirección RAM destino
; Salida:
;
; Registros afectados:
;	AF
;	HL
;—————————————————————-
oldirmv:
	push	BC		; Save counter
	ld	A, (VDPPORT1)
	ld	C, A
	inc	C
	ld	A, H
	rlc	A
	rlc	A
	and	03h	; A = página
	di		; Deshabilitamos interrupciones
	out	(C), A	; escritura de la página
	ld	A, 80h + 14	; Registro 14, acceso directo
	out	(C), A	; Escritura del valor
	ld	A, L	; 8 bits menos significativos de la dirección
	out	(C), A	; Escritura en el Puerto 1 (ya seleccionado en C)
	ld	A, H	;
	and	03fh	; Bits A13 a A8
	out	(C), A
	ei		; Rehabilitamos interrupciones
	ld	A, (VDPPORT1)	; A = n' (Puerto 0, escritura)
	ld	C, A
	pop	HL	; Restaurar el contador en HL
oldirmv_inloop:
	in	A, (C)	; VRAM -> A
	ld	(DE), A	; A -> RAM
	inc	DE	; Próximo valor
	dec	HL
	ld	A, L
	or	H	; ¿Terminado?
```

```
    jr      nz, oldirmv_inloop   ; Si no, otra vuelta
    ret
```

Queda como ejercicio para el lector (y bendita la hora ya que por primera vez he podido escribir esta frase) el crear una función que almacene una zona de memoria en otra zona de memoria de vídeo. Los cambios a realizar son mínimos. De todas maneras, hay que tener en cuenta que los parámetros de entrada deben ser compatibles con los de la función análoga proporcionada por la BIOS.

Y ahora que tenemos unas cuantas funciones básicas para comenzar a hacer nuestros pinitos con la VRAM, vamos a ver el diseño de las pantallas de nuestro editor, explicando en lo sucesivo ciertas peculiaridades y aspectos a tener en cuenta que nos será de ayuda a la hora de poder pensar en una posterior programación.



En las pantallas anteriores se observa prácticamente lo mismo. La única diferencia es que la localización de lo que yo llamo la tabla de herramientas, compuesta por la tabla de edición y la tabla de paleta, se encuentra en diferentes posiciones de pantalla. Veremos más adelante el porqué de esto. Si se presta atención, se observa como la pantalla está dividida en tres partes, justamente las tres de las que habíamos hablado con anterioridad. Además podéis observar como patrones de la primera no aparecen en la segunda ni en la tercera, y patrones de la tercera no aparecen por ejemplo en la primera, e incluso patrones de la segunda no aparecen en ninguna de las dos. Sin embargo sí hay patrones repetidos. Es decir, podemos tener 256 patrones por cada zona, algunos pueden ser los mismos que en los de las demás zonas e incluso estar en las mismas posiciones, o ser diferentes, eso ya dependerá de cómo queramos usar dichos patrones en nuestra posterior aplicación o juego.

Observando con más detenimiento, vemos que ambas 3 zonas tienen, justamente hacia el final de caza zona, una hilera de 16 patrones cada uno de un color diferente. Esto tiene una explicación. Si os fijáis en lo que anteriormente he denominado tablón de herramientas, esa misma hilera aparece pero con los colores agrupados de dos en dos. Esta disposición es la que he decidido que debía tomar la paleta de colores. Puesto que para editar un patrón necesitamos también editar los colores de éste, la paleta nos servirá para poder elegir el color correspondiente. El cómo decidimos qué color corresponde al valor *1* y qué color corresponde al valor *0* ya lo veremos más adelante.
Es por la necesidad de esta paleta, por la que tenemos que tener definidos estos 16 caracteres con esos colores, puesto que deben de ser usados para poder pintar la paleta de la tabla de herramientas. Y es necesario en los 3 tercios porque el cuadro puede estar en cualquier parte de la pantalla. Bueno, en esto último quizá os he mentido. El cuadro actualmente sólo puede tomar las 4 esquinas, con lo que los 16 patrones de color de la zona central podrían ahorrarse, pero los he dejado por si en un futuro decidiera que la tabla de herramientas se pudiera mover con desplazamientos de tamaño de carácter (de 8 pixels en 8 pixels).
Así pues, si es necesario definir estos patrones porque sino no podríamos dibujar la paleta (y después veremos que tampoco podríamos dibujar el carácter ampliado que aparece en la tabla de edición), ¿cómo vamos a editar esos 16 caracteres que por derecho propio nos pertenecen? ¿Acaso deberemos conformarnos con definir 256-16 caracteres únicamente? No. Gracias a las funciones de copia de VRAM a RAM y viceversa, podemos reservar un espacio en memoria para guardar los patrones que deberían de estar ahí. Pero entonces, ¿cuándo podremos editarlos? Pues simplemente haciendo que los 16 patrones de colores se muevan a otro lado donde no molesten. Y entonces, ¿qué ocurre con los patrones a donde van parar los 16 patrones de colores? Pues se salvan de la misma manera que se hizo con los últimos. Por simplicidad, he decidido que se podrá hacer un *swap* de los 16 patrones de colores entre el final de cada tercio o zona y el inicio de cada uno de ellos. Veamos 2 pantallas explicativas.
En las imágenes anteriores podéis observar el cambio de posición



de los patrones usados para el tablón de paleta y para el tablón de edición. Los patrones sobre los que se redefinen son salvados en RAM de manera que posteriormente pueden ser recuperados. Si prestáis algo de atención, el carácter del primer tercio con las franjas de colores rojo y amarillas había sido ocultado por los patrones de colores. Al hacer el *swap* de posición, se ha restaurado en VRAM la definición del patrón que teníamos salvada en RAM. Veamos el código para hacer el *swap* de los patrones.

```
SC2PATTERNTABLE1        .equ 00000h
SC2PATTERNNAMETABLE1    .equ 01800h
SC2COLORTABLE1          .equ 02000h

SC2PATTERNTABLE2        .equ 00800h
SC2PATTERNNAMETABLE2    .equ 01900h
SC2COLORTABLE2          .equ 02800h

SC2PATTERNTABLE3        .equ 01000h
SC2PATTERNNAMETABLE3    .equ 01A00h
SC2COLORTABLE3          .equ 03000h

PatternsSavedFirst  .db 0   ; Indica si se han salvado los 16
                             patrones iniciales de cada zona o
                             bien los finales
Patterns1Save           .ds 8*16 ; Para los 16 patrones
                             usados por la paleta
Patterns2Save           .ds 8*16 ; Para los 16 patrones
                             usados por la paleta
Patterns3Save           .ds 8*16 ; Para los 16 patrones
                             usados por la paleta
PatternsColors1Save     .ds 8*16 ; Para los colores de
                             patrones usados por la paleta
PatternsColors2Save     .ds 8*16 ; Para los colores de
                             patrones usados por la paleta
PatternsColors3Save     .ds 8*16 ; Para los colores de
                             patrones usados por la paleta

ColorTablePalette
.db 0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1
.db 2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3
.db 4,4,4,4,4,4,4,4,5,5,5,5,5,5,5,5
.db 6,6,6,6,6,6,6,6,7,7,7,7,7,7,7,7
.db 8,8,8,8,8,8,8,8,9,9,9,9,9,9,9,9
.db 10,10,10,10,10,10,10,10,11,11,11,11,11,11,11,11
.db 12,12,12,12,12,12,12,12,13,13,13,13,13,13,13,13
.db 14,14,14,14,14,14,14,14,15,15,15,15,15,15,15,15


SwapPalettePatterns:
        call    RestorePatternsReplaced
        ld      A, (PatternsSavedFirst)
        or      0
        jr      Z, SPP_SetPatternsSavedFirst
        xor     A
        ld      (PatternsSavedFirst), A
        jp      SPP_ContinueSwap
SPP_SetPatternsSavedFirst:
        ld      A, 1
        ld      (PatternsSavedFirst), A
SPP_ContinueSwap:
        call    SavePatternsReplaced
        call    CopyPalettePatterns
        ret
RestorePatternsReplaced:
        ld      A, (PatternsSavedFirst)
        or      A
        jr      Z, RPR_SavedLast
        ld      HL, 0
        jp      RPR_SavedFirst
RPR_SavedLast:
        ld      HL, 240*8
RPR_SavedFirst:
; Restaura los colores de los patrones
        push    HL
        ld      DE, SC2COLORTABLE1
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, PatternsColors1Save
        ld      BC, 8*16
        xor     A
        call    oldirvm
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE2
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, PatternsColors2Save
        ld      BC, 8*16
        xor     A
        call    oldirvm
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE3
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, PatternsColors3Save
        ld      BC, 8*16
        xor     A
        call    oldirvm
; Restaura los patrones
        pop     HL
        push    HL
        push    HL
        pop     DE
        ld      HL, Patterns1Save
        ld      BC, 8*16
        xor     A
        call    oldirvm
        pop     HL
        push    HL
        ld      DE, SC2PATTERNTABLE2
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, Patterns2Save
        ld      BC, 8*16
        xor     A
```

```
        call    oldirvm
        pop     HL
        ld      DE, SC2PATTERNTABLE3
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, Patterns3Save
        ld      BC, 8*16
        xor     A
        call    oldirvm
        ret

SavePatternsReplaced:
        ld      A, (PatternsSavedFirst)
        or      A
        jr      Z, SPR_SavedLast
        ld      HL, 0
        jp      SPR_SavedFirst
SPR_SavedLast:
        ld      HL, 240*8
SPR_SavedFirst:
; Salva los colores de los patrones
        push    HL
        ld      DE, SC2COLORTABLE1
        add     HL, DE
        ld      DE, PatternsColors1Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE2
        add     HL, DE
        ld      DE, PatternsColors2Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE3
        add     HL, DE
        ld      DE, PatternsColors3Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
; Salva los patrones
        pop     HL
        push    HL
        ld      DE, Patterns1Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
        pop     HL
        push    HL
        ld      DE, SC2PATTERNTABLE2
        add     HL, DE
        ld      DE, Patterns2Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
        pop     HL
        ld      DE, SC2PATTERNTABLE3
        add     HL, DE
        ld      DE, Patterns3Save
        ld      BC, 8*16
        xor     A
        call    oldirmv
        ret

CopyPalettePatterns:
        ld      A, (PatternsSavedFirst)
        or      A
        jr      Z, CPP_CopyLast
        ld      HL, 0
        jp      CPP_CopyFirst
CPP_CopyLast:
        ld      HL, 240*8
CPP_CopyFirst:
; Copia los colores de los patrones de la paleta
        push    HL
        ld      DE, SC2COLORTABLE1
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, ColorTablePalette
        ld      BC, 8*16
        xor     A
        call    oldirvm
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE2
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, ColorTablePalette
        ld      BC, 8*16
        xor     A
        call    oldirvm
        pop     HL
        push    HL
        ld      DE, SC2COLORTABLE3
        add     HL, DE
        push    HL
        pop     DE
        ld      HL, ColorTablePalette
        ld      BC, 8*16
        xor     A
        call    oldirvm
; Copia los patrones de la paleta. Son todo ceros, así que se
puede hacer un stosb
        pop     HL
        push    HL
        ld      BC, 8*16
        xor     A
        call    stosbvrm
        pop     HL
        push    HL
        ld      DE, SC2PATTERNTABLE2
        add     HL, DE
        ld      BC, 8*16
```

```
        xor     A
        call    stosbvrm
        pop     HL
        ld      DE, SC2PATTERNTABLE3
        add     HL, DE
        ld      BC, 8*16
        xor     A
        call    stosbvrm
        ret
```

Bueno, aquí hay bastante código por comentar. Pero en realidad es mucho menos difícil de lo que pueda parecer. El ensamblador tienen la peculiaridad de ser un lenguaje muy vertical. Cualquier cosa, por muy sencilla que sea, se convierte fácilmente en un conjunto considerable de líneas de código. Sin embargo, si pudiéramos traducir estas líneas a un lenguaje más horizontal como por ejemplo Basic, veríamos que el código generado no es tanto.

Veamos la primera función: *SwapPalettePatterns*. Esta función no tiene mucho misterio. Primeramente restauramos los patrones que habían sido machacados con definiciones de los patrones de paleta. Acto seguido se cambia una variable de tipo booleano para especificar que los patrones sustituidos van a ser los del principio o bien los del final, dependiendo del valor que tuviera antes. Después se salvan los patrones que van a ser sustituidos por los que necesitamos para la paleta de colores y por último se genera los patrones de la paleta en el lugar correspondiente.

En la siguiente función, *RestorePatternsReplaced*, se comprueba cuáles fueron los últimos patrones guardados, si los del principio o los del final. Dependiendo del resultado, HL toma el desplazamiento del principio o el final de una zona de definición de patrón o definición de color. Después este valor se usa para añadírselo a los tres desplazamientos correspondientes a cada una de las zonas de definición de color, consiguiendo con esto restaurar una tras otras las definiciones de color de cada uno de los 16 patrones que se habían salvado. Seguidamente se hace lo mismo pero con la definición de los patrones en sí. Hay que tener en cuenta que es importante no sólo haber guardado la información de los patrones si no también la información del color, porque si no gran parte de nuestro trabajo a la hora de definir los patrones y sus colores se hubiera perdido.

El procedimiento *SavePatternsReplaced* apenas merece ser comentado. Es una copia exacta de la anterior función, cambiando las etiquetas básicamente, y en lugar de copiar de memoria a memoria de vídeo, lo que se hace es copiar de memoria de vídeo a memoria.

En el último caso, *CopyPalettePatterns* lo que hace es generar los colores de los patrones que conformarán la paleta. Para ello utiliza una zona de memoria en la que están definido dichos patrones de color, que son nada más y nada menos que conjuntos seguidos de 8 bytes que toman valores desde el cero hasta el quince, todos los índices de la paleta. Puesto que al escribir estos valores en cada una de las tablas de colores, y en la situación correcta (al principio o bien al final de cada una de ellas), también es necesario escribir las definiciones de los patrones correspondientes, se procede a escribir definiciones cuyos bytes son totalmente cero.

Es decir, cada uno de los bits de cada byte es cero, con lo cual se usará el color cero de la tabla de colores para el patrón correspondiente, pero como antes habíamos definido dicha tabla del 0 al 16, cada patrón se verá en pantalla completamente de ese mismo color. Si hubiéramos querido definirlos con unos en lugar de con ceros, tendríamos que haber escrito otra tabla de colores, que sería exactamente la anterior, pero con cada uno de los bytes de que se componen multiplicados por 16 (desplazados a la izquierda 4 posiciones).

Anteriormente había comentado que el tablón de edición también necesitaba esta definición de 16 colores. Como es natural, para ver el patrón ampliado, cada uno de los píxeles se convierten completamente en un patrón cuyo color debe de ser el mismo que el del pixel del patrón ampliado. Veamos cómo hacer esta función de ampliación, junto con el dibujado de la tabla de paleta que forma parte de la tabla de herramientas.

```
PaletteBoardCurrPosX        .db 0
PaletteBoardCurrPosY        .db 0
PaletteBoardPrevPosX        .db 0
PaletteBoardPrevPosY        .db 0

PaletteBoardFirst   .db 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, 14, 15
PaletteBoardLast   .db 240, 241, 242, 243, 244, 245, 246, 247
.db 248, 249, 250, 251, 252, 253, 254, 255
PatternEditBoard
.db 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h
.db 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h
.db 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h
.db 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h
.db 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h
.db 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h
.db 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h
.db 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h, 0F4h, 0F3h

HiddenAreaSave     .ds 80


RenderScreen:
        call    WaitVblank

        ld      A, (PaletteBoardCurrPosX)
        ld      B, A
        ld      A, (PaletteBoardPrevPosX)
        cp      B
        jr      NZ, RS_RestoreAndSaveHiddenArea
        ld      A, (PaletteBoardCurrPosY)
        ld      B, A
        ld      A, (PaletteBoardPrevPosY)
        cp      B
        jr      Z, RS_NoNeedToRestoreSave
RS_RestoreAndSaveHiddenArea:
        call    RestoreHiddenArea
        call    SaveHiddenArea
RS_NoNeedToRestoreSave:
        call    DrawPaletteBoard
        call    DrawPatternEditBoard
        ret
```

```
;————————————————————————
; Calculates HL offset dado PosX and PosY en
; A, X
; B, Y
; Si HL tenía algún valor, se le añade el calculado
;————————————————————————
CalculateHLOffset:
; Calculate the coordinate (in HL)
        push    AF
        ld      A, B
        or      A
        jr      Z, CHLO_ZeroYCoord   ; ¿Se necesita hacer
                                       loop?
        ld      B, A
CHLO_CoordCalc:
        ld      A, 32
        add_hl_a
        djnz    CHLO_CoordCalc
CHLO_ZeroYCoord:
        pop     AF
        add_hl_a            ; Añadir la coordenada X
        ret

;————————————————————————
; Salva el area ocultada por la tabla de herramientas
;————————————————————————
SaveHiddenArea:
        ld      HL, SC2PATTERNNAMETABLE1
        ld      A, (PaletteBoardCurrPosY)
        ld      B, A
        ld      A, (PaletteBoardCurrPosX)
        call    CalculateHLOffset
        ld      DE, HiddenAreaSave
        ld      C, 8
SHA_Loop1:
        ld      B, 10
SHA_Loop2:
        call    onrdvrm
        ld      (DE), A
        inc     HL
        inc     DE
        djnz    SHA_Loop2
        ld      A, 32 - 10
        add_hl_a
        dec     C
        jr      NZ, SHA_Loop1
        ret
;————————————————————————
; Restaura el area ocultada por la barra de herramientas
;————————————————————————
RestoreHiddenArea:
        ld      HL, SC2PATTERNNAMETABLE1
        ld      A, (PaletteBoardPrevPosY)
        ld      B, A
        ld      A, (PaletteBoardPrevPosX)
        call    CalculateHLOffset
        ld      DE, HiddenAreaSave
        ld      C, 8
RHA_Loop1:
        ld      B, 10
RHA_Loop2:
        ld      A, (DE)
        call    onwrvrm
        inc     HL
        inc     DE
        djnz    RHA_Loop2
        ld      A, 32 - 10
        add_hl_a
        dec     C
        jr      NZ, RHA_Loop1
        ret

;————————————————————————
; Dibuja la tabla de paleta
;————————————————————————
DrawPaletteBoard:
        ld      HL, SC2PATTERNNAMETABLE1
        ld      A, (PaletteBoardCurrPosY)
        ld      B, A
        ld      A, (PaletteBoardCurrPosX)
        call    CalculateHLOffset
        ld      A, 8
        add_hl_a                    ; Relative X position of
                                      palette from X pos
        ld      A, (PatternsSavedFirst)
        or      0
        jr      Z, DPB_SavedLast
        ld      DE, PaletteBoardFirst
        jp      DPB_BeginDraw
DPB_SavedLast:
        ld      DE, PaletteBoardLast
DPB_BeginDraw:
        ld      B, 8
DPB_Loop:                   ; A loop of 8 times two colors =
                              16 colors
        ld      A, (DE)
        call    onwrvrm
        inc     HL
        inc     DE
        ld      A, (DE)
        call    onwrvrm
        inc     HL
        inc     DE
        ld      A, 32 - 2
        add_hl_a
        djnz    DPB_Loop
        ret
;————————————————————————
; Dibuja la tabla de edición
;————————————————————————
DrawPatternEditBoard:
        ld      HL, SC2PATTERNNAMETABLE1
        ld      A, (PaletteBoardCurrPosY)
        ld      B, A
        ld      A, (PaletteBoardCurrPosX)
        call    CalculateHLOffset
        ld      DE, PatternEditBoard
        ld      C, 8
```

```
DPEB_Loop1:
        ld      B, 8
DPEB_Loop2:
        ld      A, (DE)
        call    onwrvrm
        inc     HL
        inc     DE
        djnz    DPEB_Loop2
        ld      A, 32 - 8
        add_hl_a
        dec     C
        jr      NZ, DPEB_Loop1
        ret
```

El primero de los métodos copiados renderiza la pantalla. De hecho, la pantalla no tiene que redibujarse siempre, sólo cuando un cambio hace necesario que esto sea así. Pero el mover la tabla de herramienta es uno de estos cambios. Para ello, si es necesario, primeramente restaura los nombres de patrones sobre los que estaba, para posteriormente salvar los patrones sobre los cuales se va a mover. En un caso de ejemplo, si estamos en la esquina superior izquierda y nos movemos hacia la derecha, el procedimiento restaurará del espacio de memoria reservado los nombres de patrón 0 a 9, 32 a 41, etc. y salvará después los nombres de patrones del 22 al 32, del 54 al 63, etc. Acto seguido dibujará la tabla de paleta y la tabla de edición.

La siguiente función añade a HL el offset correspondiente a un nombre de patrón cuyas coordenadas de pantalla (en modo carácter) están guardadas en A y B, siendo A la coordenada X entre 0 y 31, y B la coordenada Y entre 0 y 23. Esta función es socorrida en los siguientes procedimientos.

La función de salvaguarda de la zona ocultada por la tabla de herramientas, calcula la posición donde va a ir a parar esta tabla, y empieza a salvar línea por línea cada una de las 8 líneas de 10 caracteres que serán sobrescritos, pues la tabla de herramientas cuenta de 8 líneas de 8 caracteres para la tabla de edición, y 8 líneas de 2 caracteres para la tabla de paleta.

La función de recuperación de la zona oculta es una copia de la primera, pero en lugar de hacer los movimientos de memoria de vídeo a memoria normal, lo hace en sentido inverso. Realmente no tiene mucho más misterio que la función predecesora.

Para dibujar la paleta, tenemos que tener en cuenta si los patrones que definen los colores de paleta se hallan definidos al principio o al final de cada zona. Es lo primero que se comprueba en este método de dibujado de la tabla de paleta, para acto seguido dibujar cada una de las 8 líneas de 2 caracteres. Hay que tener en cuenta que no empieza a dibujar desde la coordenada X de la posición del tablón, si no desde la coordenada X + 8, para no sobrescribir la tabla de edición.

La función de dibujado de la tabla de edición de patrón escribe en memoria de vídeo las 8 líneas de 8 caracteres de los que se compone, y que están previamente definidos en memoria. El proceso por el cual esta zona de memoria con la información del carácter editado se actualiza, se verá a continuación. Es un proceso que puede parece un tanto complicado de entender, pero en definitiva no es más que ponerse a pensar en lo datos que necesita la VRAM que le digamos para dibujarlo, y en los datos que necesitamos nosotros obtener para generar los primeros. La función es la que se muestra a continuación:

```
SelectedPatternLineColor0     .db 0
SelectedPatternLineColor1     .db 0
SelectedPattern               .ds 8
SelectedPatternColors                 .ds 8
SelectedPatternAddress                .dw 0
SelectedPatternColorAddress   .dw 0

;————————————————
; Actualiza la tabla de edición con el carácter que hay en PosX
PosY
; A = Pos X (0, 31)
; B = Pos Y (0, 23)
;————————————————-

UpdatePatternEditBoard:
        ld      HL, 0
        call    CalculateHLOffset
; Tenemos que hacer HL * 8 (desplazar a la izquierda 3 bits)
        sla     L
        rl      H
        sla     L
        rl      H
        sla     L
        rl      H

;       ld      DE, SC2PATTERNTABLE1
;       add     HL, DE
; Guardar la dirección del patrón seleccionado para cuando se
modifique
        ld      (SelectedPatternAddress), HL
        ld      DE, SelectedPattern
        xor     A
        ld      BC, 8
        call    oldirmv         ; Obtiene el patrón selec
                                  cionado (8 bytes)
        call    UpdatePatternEditColors
        ret

UpdatePatternEditColors:
        ld      DE, PatternEditBoard
        ld      HL, SelectedPattern
        ld      IX, SelectedPatternColors
        ld      C, 8
UPEBC_MapColorsToPatterns2:
        ld      B, 8
; Aquí sería buen momento para obtener el color de "1" y "0"
para cada líneas
        ld      A, (IX + 0)     ; Información del color de
                                  la línea
        and     00Fh
        ld      (SelectedPatternLineColor0), A
        ld      A, (IX + 0)
        and     0F0h
        srl     A
```

```
        srl     A
        srl     A
        srl     A
        ld      (SelectedPatternLineColor1), A
        ld      A, (HL)                 ; Obtener una línea del
                                        patrón seleccionado
UPEBC_MapColorsToPatterns1:
        rlc     A
        jp      C, UPEBC_IsOne          ; Bit a 1, se debe selec-
cionar el color para "1"
        push    AF
        ld      A, (PatternsSavedFirst) ; Si los patrones
                                        de color están al
                                        final debemos sumar
                                        240 al valor
        or      0
        jr      NZ, UPEBC_UsingFirstPatternsZero
        ld      A, (SelectedPatternLineColor0)
        or      0F0h
        jr      UPEBC_UsedLastPatternsZero
UPEBC_UsingFirstPatternsZero:
        ld      A, (SelectedPatternLineColor0)
UPEBC_UsedLastPatternsZero:
        ld      (DE), A
        pop     AF
        jr      UPEBC_WasZero
UPEBC_IsOne:
        push    AF
        ld      A, (PatternsSavedFirst)
        or      0
        jr      NZ, UPEBC_UsingFirstPatternsOne
        ld      A, (SelectedPatternLineColor1)
        or      0F0h
        jr      UPEBC_UsedLastPatternsOne
UPEBC_UsingFirstPatternsOne:
        ld      A, (SelectedPatternLineColor1)
UPEBC_UsedLastPatternsOne:
        ld      (DE), A
        pop     AF
UPEBC_WasZero:
        inc     DE
        djnz    UPEBC_MapColorsToPatterns1
        inc     HL
        inc     IX
        dec     C
        jr      NZ, UPEBC_MapColorsToPatterns2
        ret
```

A pesar del código anteriormente mostrado, voy a intentar explicar lo que hago, de forma más o menos plana, y el lector será quien intente identificar cada una de las partes. Quizá más de uno esté pensando que soy más perezoso como redactor que como programador, y quizá no le falte razón, pero también es una buena excusa para que el propio lector, y sobre todo el que tiene más dificultades a la hora de entender ensamblador, ejercite un poco su mente y sea capaz de extraer las piezas del rompecabezas para entender el conjunto completo.

¿Qué es lo que necesitamos decirla a la VRAM para que dibuje un carácter cualquiera como si estuviese ampliado? Un carácter ampliado, en nuestro editor, es representado por un conjunto de líneas de 8 caracteres de tipo patrón de paleta, así pues, cada uno de estos caracteres representará exactamente un pixel del carácter que queremos dibujar ampliado. Puesto que cada carácter de los que se compone el carácter ampliado corresponde con uno de los patrones de paleta, sabemos que esos caracteres tomarán valores de 0 a 15, si los patrones de paleta están definidos al inicio de caza zona, o bien de 240 a 255, si están definidos al final. Por simplicidad, vamos a suponer que se hallan definidos al principio de cada zona. Para crear esa información de caracteres que representan pixeles, necesitamos sin duda la información del color del carácter que se va a ampliar. Tenemos una zona de memoria donde guardaremos esa información. Una vez obtenida esa información, sabemos, para cada línea del patrón del carácter, qué color ha de tomar el pixel cuyo valor sea 1, y qué color ha de tomar el pixel cuyo valor sea 0, pero no tenemos información sobre los valores de ceros y de unos del patrón. Es pues, otro dato de que debemos extraer, y que también almacenamos en memoria para su posterior uso.

El algoritmo para crear la información del carácter ampliado es el siguiente. Por cada línea del carácter, leemos la información del patrón a la vez que la información de los colores que toma el 0 y el 1 en esa línea. Dicha información sobre los colores la almacenamos en unas variables que nos serán de utilidad. Para cada bit del byte del patrón, que corresponde a una línea del carácter, miramos qué valor toma. Si toma valor 0, entonces almacenamos en la zona de memoria del carácter ampliado, correspondiente a ese bit (o pixel a efectos gráficos), el índice a la paleta correspondiente al color del valor 0. Si es uno, hacemos lo propio. Si esto lo hacemos para cada línea, obtendremos la información para nuestro carácter ampliado.

Si los patrones de los colores de paleta se hubieran encontrado al final de cada zona en lugar del principio, simplemente tendríamos que recorrer la información del carácter ampliado y sumarle 240. Yo lo hago directamente para cada byte del carácter ampliado en lugar de esperarme al final, pero el resultado obtenido es el mismo, y hasta incluso menos complicado a la hora de leer código.

Bien, ya tenemos nuestra pantalla dibujada, la posibilidad de ver todos los caracteres, cambiar los patrones de paleta de lugar para que no nos molesten, de dibujar los caracteres ampliados para poder tener mejor control a la hora de editarlos, y con un poco de código más, mover la tabla de herramientas donde queramos (esto no se explicará pero también se halla implementado). Ahora nos falta lo más importante, y lo que da nombre al editor, y es precisamente **editar** el carácter que hemos seleccionado (Yo lo hago moviendo un sprite cursor y calculando la posición en la que se encuentra). Esta es la parte que vamos a ver a continuación. Como siempre, el código precederá a la explicación.

```
ShiftPressed            .db 0
SpacePressed            .db 0
CursorPosX              .db 0
CursorPosY              .db 0
CursorMinX              .db 0
CursorMaxX              .db 31
CursorMinY              .db 0
```

```
CursorMaxY                  .db 23
WorkingWithColorZero        .db 0
LineBeignEdited   .db 0
ColumnBeignEdited .db 0
SelectedColorAsOne          .db 0
SelectedColorAsZero         .db 0

EditModeManagement:
        ld      A, (ShiftPressed)
        or      A
        jr      Z, EMM_NoPaintOrColorSelForZero
        ld      A, 1
        ld      (WorkingWithColorZero), A
        jp      EMM_UpdateColorAndPattern
EMM_NoPaintOrColorSelForZero:
        xor     A
        ld      (WorkingWithColorZero), A
        ld      A, (SpacePressed)
        or      A
        jp      Z, EMM_NoPaintOrColorSelection
EMM_UpdateColorAndPattern:
        ld      A, (CursorMaxX)
        dec     A
        ld      B, A
        ld      A, (CursorPosX)
        cp      B
        jp      P, EMM_IsInsidePaletteBoard
; Dibujar "1" y actualizar el color "1" de los colores del patrón
editado
        ld      A, (CursorMinX)
        ld      B, A
        ld      A, (CursorPosX)
        sub     B
        ld      (ColumnBeignEdited), A      ; Guarda la X
                                             relativa a la
                                            tabla de edición
        ld      A, (CursorMinY)
        ld      B, A
        ld      A, (CursorPosY)
        sub     B                   ; A tiene la Y relativa a la
                                     tabla de edición que es la
                                     línea a editar
        ld      (LineBeignEdited), A
        ld      A, (WorkingWithColorZero)
        or      A
        jr      NZ, EMM_UpdateZeroColor
; Actualizar color para "1"
        ld      A, (SelectedColorAsOne)
        sla     A
        sla     A
        sla     A
        sla     A
        ld      B, A
        ld      A, (LineBeignEdited)
        ld      HL, SelectedPatternColors
        add_hl_a
        ld      A, (HL)
        and     00Fh
        jp      EMM_UpdateColor
EMM_UpdateZeroColor:
; Actualizar color para "0"
        ld      A, (SelectedColorAsZero)
        ld      B, A
        ld      A, (LineBeignEdited)
        ld      HL, SelectedPatternColors
        add_hl_a
        ld      A, (HL)
        and     0F0h
EMM_UpdateColor:
        or      B                   ; A tiene el nuevo byte de
                                    color para la línea editada
        ld      (HL), A             ; Actualizar también el c
                                    olor en memoria
                                   ;(para la tabla de edición )
        push    AF                  ; Salva información del
                                    color
        ld      HL, (SelectedPatternColorAddress)
        ld      A, (LineBeignEdited)
        add_hl_a                    ; HL tiene el byte del
                                    color a modificar
        pop     AF                  ; Recupera la información
                                    del color
        call    onwrvrm             ; Escribirlo en vram

; Ahora tenemos que cambiar el valor X("0" or "1") por "1" or
"0"
        ld      A, (ColumnBeignEdited)
        inc     A
        ld      B, A                ; Preparar B
        xor     A
        scf
EMM_ColumnShift:
        rr      A
        djnz EMM_ColumnShift        ; Obtener una máscara de
                                    bit correcta
        push    AF
        ld      A, (WorkingWithColorZero)
        or      A
        jr      NZ, EMM_SetPatternBitToZero
        pop     AF
        ld      B, A
        ld      HL, (SelectedPatternAddress)
        ld      A, (LineBeignEdited)
        add_hl_a
        call    onrdvrm             ; A tendrá el byte editado
        or      B                   ; Or, luego, escribimos un
                                     1
        jp      EMM_SetOrResetPatternBit
EMM_SetPatternBitToZero:
        pop     AF
        cpl
        ld      B, A
        ld      HL, (SelectedPatternAddress)
        ld      A, (LineBeignEdited)
        add_hl_a
        call    onrdvrm             ; A tendrá el byte editado
        and     B                   ; And tras cpl, luego,
                                     escribimos un 0
EMM_SetOrResetPatternBit:
        call    onwrvrm             ; Actualizar el valor
```

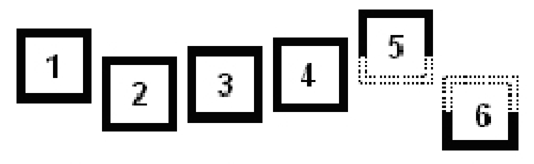
```
        ld      HL, (SelectedPatternAddress)
        ld      DE, SelectedPattern
        xor     A
        ld      BC, 8
        call    oldirmv             ; Obtiene el patrón editado
        call    UpdatePatternEditColors  ; Actualiza colores
                                         del tablón de
                                         edición
        ret
EMM_IsInsidePaletteBoard:
        sub     B           ; A tendrá 0 or 1 (indica la columna
                              de la tabla de paleta)
        ld      B, A
        ld      A, (CursorMinY)
        ld      C, A
        ld      A, (CursorPosY)
        sub     C           ; A tendrá la linea relativa a la tabla
                              de paletasla A
        add     A, B        ; A obtendrá el color de la paleta
        push    AF
        ld      A, (WorkingWithColorZero)
        or      A
        jr      NZ, EMM_SetSelectedColorAsZero
        pop     AF
        ld      (SelectedColorAsOne), A
        ret
EMM_SetSelectedColorAsZero:
        pop     AF
        ld      (SelectedColorAsZero), A
EMM_NoPaintOrColorSelection:
        ret
```

Primeramente se comprueba si las variables que indican si se ha apretado espacio o bien shift están activadas, de lo contrario se evita hacer ningún proceso. He elegido el espacio para trabajar con el valor 1 de los patrones, y el shift para trabajar con el 0. Una vez alguno de ellos se ha activado, se comprueba si el cursor se haya en la zona de la tabla de paleta o la tabla de edición. Para ello se comprueba el valor de la posición del cursor con los valores máximos y mínimos que puede tomar dentro de la zona de la tabla de herramientas, valores que se actualizan convenientemente según la posición de dicha tabla.
Si la pulsación ha caído dentro de la tabla de paleta, se selecciona el color correspondiente y se almacena en la variable de color elegido como cero, o bien como uno, dependiendo de la tecla que se haya apretado. Si la tecla se ha apretado dentro de la tabla de edición, entonces se procede a modificar el carácter, pues se está editando.
Para ello debemos almacenar ciertos valores que nos serán de utilidad más adelante, y que son la línea del carácter que se está editando, y la columna del mismo. Una vez hecho esto, calculamos el offset del byte de la tabla del color del carácter correspondiente a la línea que se está editando, y modificamos el índice del color correspondiente al valor 1 o bien al 0 dependiendo de cuál estemos editando. Así pues, ya tenemos una cosa hecha, ahora sólo falta modificar el patrón en sí. Otra vez necesitamos la variable que guarda la línea editada para calcular el offset al byte del patrón correspondiente cuya información tiene los datos sobre la línea que está siendo editada. Ahora debemos o bien cambiar algún cero por un uno, lo cual implicará una operación lógica **or**, o bien deberemos cambiar un uno por un cero, con lo que necesitaremos realizar una operación **and**. La información sobre el bit que debemos cambiar la podemos extraer de la columna que está siendo editada. Así pues, con esta información nos creamos un byte en el que en la columna editada el valor del bit toma justamente el valor 1. Así pues, si debemos cambiar un cero por un uno, simplemente tenemos que hacer el or de ese byte con el byte del patrón correspondiente a la línea editada y que ya teníamos antes. En el caso de necesitar cambiar el valor por un cero, debemos de complementar el byte y hacer un and. Como veis, explicado no es tan difícil, y en definitiva, programarlo después tampoco lo es tanto, simplemente hay que seguir paso a paso nuestras palabras y transformarlas a código. Evidentemente en este proceso puede haber errores, pero normalmente acabaremos saliéndonos con la nuestra.

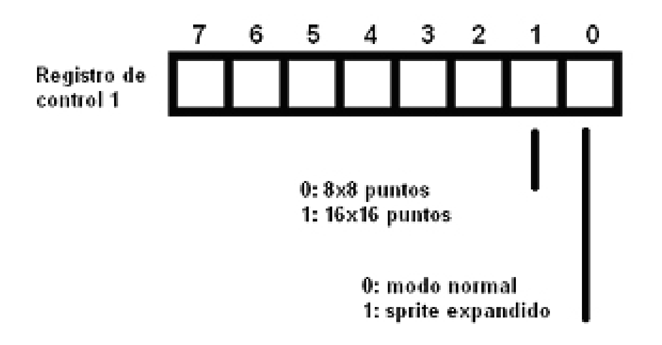
En este momento ya tenemos las partes más importantes de nuestro programa, si no contamos las funciones para grabar a disco, que aunque implementadas no las comentaré por irse un poco del tema. Sólo faltaría hablar del control del teclado para realizar el movimiento del cursor alrededor de la pantalla, pero esto es algo que también se sale del estricto tema que se relaciona con el screen 2. Lo que sí que tiene que ver con este modo, y con otros modos en general, es el dibujado del sprite que hace de cursor. Para ello no solamente necesitamos definir el sprite, sino que debemos modificar los atributos correspondientemente para que se posicione en el lugar deseado. Eso lo veremos a continuación, pero antes decir también que existen unos comandos de copy/cut y paste que no voy a explicar porque sería repetirme una y otra vez sobre el mismo tema, copiar de VRAM a RAM y viceversa, y ya hemos visto suficientes ejemplos en este documento. Un buen ejercicio para el lector sería implementar una de estas rutinas y compararlas luego con el código proporcionado. Es probable que incluso se lleve una sorpresa al ver que la ha realizado incluso de forma más optima, y es que mi intención no era optimizar el código, si no hacerlo de la forma más entendedora posible. He primado el uso de variables al de registros por ser una forma más intuitiva y aproximada a los lenguajes de alto nivel, en los que se trabaja siempre con variables, otorgando nombres a éstas lo suficientemente explicativos como para que el lector no tenga demasiados problemas para saber qué función tiene esa variable en el código. Lo mismo he intentado hacer con el nombre de las funciones y etiquetas, cuya nomenclatura es la de tomar las siglas del nombre de la función seguido por un guión y el nombre de la etiqueta en sí, lo cual, además de saber que es de una función concreta, evita la repetición de las etiquetas, y tener que elegir nombres como etiqueta1, etiqueta2, etc. que he visto por muchos lugares y que no aclaran absolutamente nada.

Hablemos pues un poco de los sprites. Existen dos modos de sprites, pero el segundo únicamente es posible usarlo a partir de los MSX de segunda generación, por lo que nos ceñiremos exclusivamente a tratar el modo uno. En este pueden coexistir en la pantalla hasta 32 sprites, numerados del 0 al 31. Cada uno de estos sprites se dispone en un plano, y los planos con numeración menor tienen prioridad sobre los planos con numeración mayor. Cuando queremos situar los sprites en una misma línea, hasta cuatro sprites se pueden mostrar simultáneamente sin tener ningún tipo de conflicto entre ellos. A partir del

quinto sprite, y siempre que este sea de prioridad inferior al resto, las líneas de este quinto sprite que coincidan con las de los cuatro anteriores, no podrán ser mostradas. Veámoslo gráficamente.



Los sprites pueden tener dos tamaños diferentes. Pueden ser de 8 por 8 puntos, que es el modo que está seleccionado por defecto, o bien 16 por 16 puntos. Para cambiar de un modo a otro, hay que especificar el modo en el regristro de control 1 del procesador de vídeo (VDP). Además, usando el mismo registro, se puede especificar un modo de ampliación del sprite, en el que cada punto del sprite se dibuja como si fueran en realidad un cuadrado de dos puntos horizontales por dos verticales, con lo cual se obtiene un sprite del doble de tamaño, como si se hubiera hecho un zoom por dos.



A pesar de que sólo es posible mostrar simultáneamente 32 sprites, pueden tenerse definidos un total de 256 sprites en el caso de que sean de 8 por 8 puntos, o bien 64, si los sprites tienen un tamaño de 16 por 16. Cuando el modo 16x16 está seleccionado, para un único sprite se usan cuatro patrones de 8x8, que se encuentran contiguos (uno seguido de otro) en memoria de vídeo.

La tabla generadora de patrones de sprites comienza en la posición 3800h y finaliza en 3fffh, y con las mismas rutinas de escritura en la VRAM que se han visto anteriormente se pueden definir patrones de sprites de la misma forma que se hacía con los patrones de caracteres. Existe otra zona de la memoria de vídeo, la comprendida entre 1b00h y 1b7fh que recibe el nombre de tabla de atributos de sprite. En esta zona se especifican, entre otras cosas, la posición vertical y horizontal de cada sprite. Para cada uno de los 32 sprites que es posible mostrar, existen 4 bytes para especificar su atributos y que se encuentran contiguos.

Primer byte: Indica la coordenada vertical (Y) del sprite. Hay que tener en cuenta algo muy importante, y es que la posición de arriba del todo del sprite es la 255 y no la 0. Así pues, si el valor que ponemos en este byte es 0, en realidad estaremos poniendo el sprite en la coordenada vertical 1. Otro aspecto a destacar es que si se escribe el valor 208 en esta posición, el sprite no será mostrado en pantalla.

Segundo byte: Aquí se ha de escribir la coordenada horizontal (X) en la que queramos que vaya a parar el sprite. En este caso todo funciona normalmente, el 0 es el 0 y el 255 es el 255, así que no debemos preocuparnos de hacer ninguna corrección de la posición de nuestro sprite a la hora de especificar esta coordenada. Me gustaría aclarar, que tanto la coordenada X como Y son coordenadas de pixeles de pantalla, y no de caracteres. No quisiera que el lector entrase en una confusión al haber estado anteriormente hablando de coordenadas de carácter que iban de 0 a 31 para la X y de 0 a 23 para la Y.

Tercer byte: Aquí se especifica el patrón de sprite que queremos usar.

Cuarto byte: Los cuatros bits menos significativos se usan para especificar el índice del color de la paleta del cual queramos que se dibuje el sprite. Eso nos limita evidentemente a que el sprite sólo pueda tener un color, que es el que toma el valor 1. El valor cero toma el color transparente, es decir, no se dibuja. En los MSX de segunda generación se pueden especificar los colores por línea tal aunque sólo para caso en el que el bit que esté a 1, pero tendremos que conformarnos con un solo color si hacemos uso de MSX de primera generación o bien queremos hacer software compatible para este hardware. El bit más significativo se usa para hacer que el sprite sea desplazado 32 bits hacia la izquierda a la hora de ser dibujado, esto permite hacer que veamos el sprite aparecer progresivamente de la izquierda de la pantalla, en lugar de un modo brusco, pero en nuestro editor no vamos a usar esta funcionalidad.

Puesto que este es un editor simple en el que no se hace un uso exhaustivo de sprites como pudiera ser un vídeojuego, terminaré aquí la explicación sobre sprites, dejando para quizá un próximo artículo el conflicto entre sprites. Quizá el lector pueda pensar que el título del artículo no se adapte totalmente al contenido del mismo, pues hay ciertas cosas que no se han explicado, pero en el caso por ejemplo de los sprites, lo mismo valdría para screen 2 que para screen 1, con lo cual no es específico de este modo de vídeo y preferiría tratarlo de forma separada.

Bien, hasta aquí el final de la explicación del modo gráfico screen 2. Espero que hayáis podido aprender algo interesante y que os sirva para vuestros propios proyectos. Y si el editor también os gusta y lo usáis, el que os escribe se sentiría muya agradecido de verse en los créditos de vuestras creaciones como parte de los agradecimientos. Que lo programéis bien, y si algún concepto se os atraganta, no os preocupéis que el tiempo hará su trabajo para que os acabe de entrar. Un saludo y hasta el próximo número.

David Lucena

ENCUENTRO DE MSX EN OSS (Países Bajos)

17 de enero 2004

El 17 de enero de este año el MSX-NBNO, un grupo de usuarios activos de la provincia de Noord Brabant (al sur de los Países Bajos), organizó un encuentro de MSX. Nosotros vivimos en el norte, pero no dudamos en madrugar con tal de asistir a uno de los tres más importantes aconte-cimientos sobre MSX que ha logrado sobrevivir al siglo XXI en nuestro país. Llegamos sobre las 10:30 horas y lo primero que nos llamó la atención fue que no había que pagar entrada, era una feria completamente gratuita. Esto nos sorprendió, ya que parecía ser un acto bastante concurrido y amplio. Además, el viejo teatro donde se celebraba, confería al lugar una atmósfera realmente acogedora.

Nada más entrar en el salón, nos encontramos con un grupo de usuarios expertos en electrónica que habían conseguido conectar un MSX a un monitor TFT, cosa no muy fácil para la gente corriente. Este grupo mostraba en las pantallas planas geniales gráficos de juegos, sacándolos directamente del MSX.

El siguiente stand pertenecía a un grupo coleccionista de software, que traía una gran colección de juegos MSX para vender, y también algunos juegos para otros sistemas antiguos.

Otro stand que nos llamó la atención fue uno en el que se dedicaban a destrozar un Commodore 64, cosa que nos traía a la memoria la antigua "guerra de sistemas" Commodore vs. MSX. Sin embargo ya no es lo mismo, los tiempos han cambiado e incluso aquí, en los Países Bajos, ya se habla de organizar encuentros de ordenadores retro en general, abierto a todo tipo de sistemas. Así que la guerra va llegando a su fin.

Por supuesto, el *MSX Resource Center* (los encargados de mantener la www.msx.org) también pusieron su stand, donde vendían algunos juegos originales japoneses recientemente importados y ofrecían información sobre su página web y otras actividades. También se podía hojear la MSX Magazine 2003 y la MSX Magazine 2, junto con otras publicaciones sobre MSX publicadas recientemente en revistas niponas.

Otro stand de gran relevancia fue el de la *Sunrise Foundation*. Ellos suelen visitar estas ferias para vender todo tipo de software y hardware. En esta ocasión traían el CF-IDE Adapter y el CF-IDE Interface. También era posible reservar cartuchos Moonsound, aunque todavía no se conoce fecha de fabricación para éstos... una verdadera lástima. También disponían de software y hardware.

Pero la mayor atracción de la feria fue la presentación de uno de los juegos más esperados por los usuarios de MSX: ***Bombaman***, del *Team Bomba*. Muchos de los miembros del equipo acudieron al acto. Este juego está siendo distribuido por *Sunrise* y parece que está resultando todo un éxito. Es lógico, es un juego muy bueno, que merece la pena, siendo comparable en calidad a los grandes juegos que aparecieron para nuestro querido MSX durante los 80 y principios de los 90. En el viejo teatro también era posible tomarse un café o cualquier tipo de bebida. A mí el lugar me pareció ideal para este tipo de acto: el encuentro de Oss recordaba a las grandes ferias que se celebraban durante la primera mitad de los 90, pero en pequeña. Lo triste del asunto es que el viejo teatro va a ser demolido este año, y si los MSX-NBNO quieren montar la Oss 2005, tendrán que buscar un sitio nuevo. En mi opinión, la visita valió realmente la pena. A este encuentro acudieron muchos de los "personajes" del MSX, incluida gente de fuera del país. Toda la feria en conjunto consiguió ofrecernos una atmósfera muy retro. Creo que la de Oss merece ser destacada por igual junto a las ferias de Bussum y Tilburg en los Países Bajos, que también he visitado estos últimos años. Si por casualidad os encontráis por los alrededores de los Países Bajos, no dudéis en visitar una feria de MSX, ¡os gustará!

Jan Lukens

Traducción: Eva Molina









Web de productos ASCII traducida

https://ssl.ascii-store.com/corner/msx.cgi

Pág. 1
1- Página de información para encargos y reservas
¡Los cartuchos de MSX y MSX2 reviven en tu ordenador!
MSX GAME READER
¡El retorno del estándar! Un nuevo ordenador con MSXPLAYer integrado
MSXPC 20th ANNIVERSARY MODEL

2- Al fin ha comenzado la venta al público de estos productos. Aunque ha sido el inmenso apoyo recibido en los últimos años por los usuarios de MSX lo que nos ha impulsado a crearlos, hemos advertido un cierto número de cancelaciones por parte de personas que ya habían confirmado su reserva. Por lo tanto la venta al público tendrá lugar con un número limitado de unidades.

3- El plazo para efectuar la reserva concluirá en cuanto se haya alcanzado ese número concreto de unidades, por lo que les rogamos que se apresuren a formalizar su solicitud.
*Les rogamos que antes de efectuar su reserva lean cuidadosamente las notas al pie de esta página.

4- ¿Qué es el MSX Game Reader?
Es un lector de cartuchos ROM que permite ejecutarlos desde un ordenador personal, al que se conecta a través de un puerto USB. Está concebido como un complemento al emulador MSXPLAYer, gracias al que se puede revivir en la pantalla del PC la emoción de los viejos juegos de MSX. Si se conectan dos unidades al PC se pueden aprovechar las peculiaridades que brindan algunos juegos en dos cartuchos.
*El MSX Game Reader es compatible con los cartuchos diseñados para MSX y MSX2, a excepción de módems, tarjetas de sonido y otros cartuchos de expansión. Próximamente se publicará una lista de aquellos programas cuya compatibilidad se haya comprobado.

5- ¿Qué es el MSXPC 20th Anniversary Model?
Es un ordenador tipo PC con el teclado integrado en la unidad principal que recuerda en su aspecto a los antiguos MSX. La potencia de su procesador Celeron a 2,4 GHz es más que suficiente para ejecutar aplicaciones comerciales, así como el emulador MSXPLAYer sin ningún tipo de problema. Cuenta además con puertos de expansión IEEE1394, USB2.0, LAN, serie, paralelo y el interface Legacy. Sus reducidas dimensiones lo convierten en un ordenador más práctico si cabe, porque no ocupa lugar y sólo basta conectarlo a un monitor para utilizarlo sin más. Por sus características recomendamos incluso su adquisición como segundo ordenador.

6- MSX Game Reader
¡Los cartuchos de MSX reviven en tu ordenador! Con el lector de cartuchos MSX Game Reader y el emulador MSXPLAYer tendrás la oportunidad de disfrutar con aquellos juegos que no podías ejecutar a falta de un ordenador MSX.
[Especificaciones]
· Lector de cartuchos de juego MSX a través del puerto USB, concebido para su uso exclusivo con MSXPLAYer
· Compatible con USB 1.1/2.0, conector USB tipo B
· Alimentación suministrada a través del cable USB
· 1 slot, protegido por una cubierta de apertura manual
· LEDs de alimentación y de acceso incluídos
Precio: 12500 yenes (impuestos y gastos de envío aparte)
Los gastos de envío especial ascienden a 800 yenes.

7- MSXPC 20th Anniversary Model
Un ordenador con el teclado integrado en la unidad principal y MSXPLAYer preinstalado, con el que se puede reproducir toda la emoción de los juegos clásicos del MSX. Y no sólo para jugar sino también para aplicaciones más serias se trata de un PC de lo más potente. Su formato reducido y compacto lo hace muy recomendable también para la oficina.
[Especificaciones]
· CPU: Intel Celeron 2,4 GHz / Memoria: 256MB DDR.SDRM (PC2700)
· HDD: 40 GB / Unidad óptica: CD-ROM x48 / FDD 3,5" (2 modos)
· SO de serie: WindowsXP HomeEdition
· Chipset: SiS651/962 / Aceleración de gráficos: Interna
· El monitor se vende aparte
· Para la información concerniente a la placa conmemorativa será necesario contactar a través de correo eléctronico o la página web.
Precio: 98000 yenes (impuestos y gastos de envío aparte)
Los gastos de envío especial ascienden a 1000 yenes.

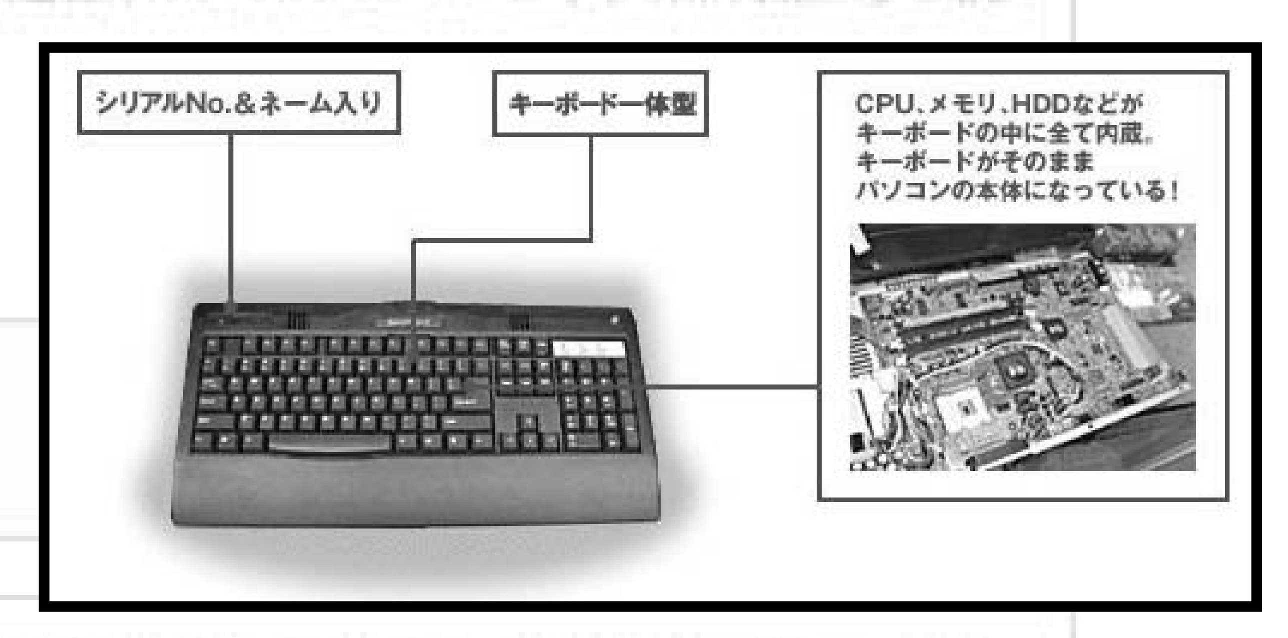
Pág. 2
1- ¡ATENCIÓN!
· Antes de efectuar su reserva deberá leer y aceptar las condiciones contenidas en el documento adjunto.
· La fecha de puesta a la venta puede sufrir alteraciones.
· El pago puede realizarse mediante tarjeta de crédito o por contrareembolso. La segunda opción comporta un incremento en los gastos de envío.
· El proceso de venta finalizará cuando se agote el número limitado de unidades.
[CONDICIONES] *(enllaç)*
[Acerca de la cancelación de pedidos]
En caso de cancelar una reserva confirmada se cargará una cantidad fija en concepto de penalización.
[Acerca de la entrega en caso de pedido múltiple]
En caso de que se adquiera algún producto que ya se halle a la venta, la entrega conjunta del pedido se realizará cuando se encuentren disponibles también los otros productos.

Pág. 3
1- El lector de cartuchos en detalle
Proyecto / Desarrollo: ASCII K.K., MSX Association
Fabricación: ASCII Solutions K.K.
Distribución: Edge K.K.
12500 yenes (impuestos aparte)
Fecha de lanzamiento: 16 de marzo de 2004

2- [Especificaciones]
· Lector de cartuchos de juego MSX a través del puerto USB, concebido para su uso exclusivo con MSXPLAYer
· Compatible con USB 1.1/2.0, conector USB tipo B
· Alimentación suministrada a través del cable USB
· 1 slot, protegido por una cubierta de apertura manual
· LEDs de alimentación y de acceso incluídos
· Incluye el logotipo oficial de MSX Game Reader
· No se suministra el cable USB. Hay que adquirir el cable (de tipo USB B – USB A) por separado.
· No se suministra manual de instrucciones. Las aclaraciones sobre el modo de uso se incluyen en el segundo número de MSX Magazine.
· Se ha comprobado el funcionamiento de los cartuchos de Konami provistos del chip de sonido SCC.
· Si se adquieren dos unidades, se pueden emplear como dos slots independientes (la conmutación entre slot 1 y slot 2 se realiza a través de interruptores).
· No se garantiza el funcionamiento a través de un hub USB. Rogamos que conecten el aparato directamente al puerto USB del ordenador.
El diseño, tamaño y peso de la unidad puede sufrir alteraciones.

3- [Advertencias]
· El lector de cartuchos MSX es un periférico desarrollado para utilizar los cartuchos MSX en entorno Windows. Aquellas personas que no dispongan de cartuchos de juego MSX no podrán emplear este aparato.
· Se necesita una máquina provista de conector USB y sistema operativo Windows (98/98SE/Me/2000/XP). No funciona en Linux ni en Mac OS.
· El producto no se acompaña del software necesario para ejecutar los cartuchos de juego ni de los drivers necesarios para que la máquina Windows lo reconozca como un periférico USB. Como software de uso se necesita la versión de MSXPLAYer distribuida conjuntamente con el segundo número de MSX Magazine; por lo que respecta a los drivers, se pueden descargar desde la página web.
· El lector de cartuchos no es compatible con generadores de sonido FM, modems y demás cartuchos de expansión. Tampoco se garantiza la plena compatibilidad con todos los cartuchos de juego MSX.
· Se puede dar el caso de que un mal estado de conservación de los cartuchos impida su correcta ejecución.

Pág. 4
1- El MSXPC 20th Anniversary Model en detalle
[MSXPC 20th Aniversary Model]
98000 yenes (impuestos aparte)



Fecha de lanzamiento: 18 de marzo de 2004

2- [Tabla de especificaciones]
CPU:I ntel Celeron 2.4 GHz
Placa Base Tipo SS4SB

CPUs compatibles:
Intel Pentium 4 1,7 GHz ~ 2,66 GHz
Intel Celeron 1,7 GHz ~2,4 GHz
(Williamette and Northwood 478)

Chipset: SIS651

FSB: 400/533 MHz

LAN: Realtek 10/100Mbps Ethernet Onboard
Tarjeta de sonido: AC97 3D Sound (montada en la placa)
VGA:VIA Apollo PEL133 Interna
Modos de visualización a 16 millones de colores (1280x1024, 1024x768, 800x600, 640x480)
Memoria: 256MB DDR-SDRAM (2 slots de DDR SDRAM PC2700: posibilidad de aumentar la memoria hasta un máximo de 2GB)
Disco Duro: 40GB 5400rpm ATA-100
Floppy: 2Model 1,44MB FDD
Unidad óptica: Lector de CD-ROM de 24 velocidades
Slots de expansión y puertos de E/S:
PCI: 0
AGP: No disponible
USB: 4 puertos USB2.0 (2 laterales y 2 traseros)
IEEE1394:2

Serie:1 trasero
Paralelo: 1 trasero
VGA: 1 trasero
Audio: Mic, Line in, Line out
Joystick: No disponible
LAN: 1 trasero

BIOS: AwardBIOS
Periféricos suministrados: Ratón PS2 con rueda de desplazamiento

Sistema operativo: Microsoft WindowsXP Home
Garantía Reparación por un año, con portes de envío gratuitos
Dimensiones: Ancho, 459mm / Largo, 228 mm / Alto, 59 mm
Peso: 3,4 Kg aprox.
Extras: Emblema. Placa metálica con el nombre y número de serie grabados
Software incluido: Hydlide, Hydlide II, Hydlide III, Wakusei Mefius, Relics, MSXPlayer (handle version)

Los datos marcados en rojo están actualizados a 26 de diciembre. El teclado es de distribución japonesa.

Pág. 5
1- [Advertencias]
No se suministra unidad de visualización, que deberá ser adquirida aparte.
· No se suministra software de aplicación (tipo MS Office).
· El periodo de garantía se extenderá por un año a partir de la fecha de entrega.
·No se puede utilizar el slot PCI (reservado en exclusiva para la unidad óptica)

Pág. 6
1- Les rogamos que antes de efectuar su pedido lean con detenimiento y acepten las siguientes condiciones.

2- [Acerca del lector de cartuchos de juego MSX a través de puerto USB para uso exclusivo con MSXPLAYer (MSX Game Reader)]
Detalles: Lector de cartuchos de juego MSX a través de puerto USB para uso exclusivo con MSXPLAYer (versión Windows)
· Drivers incluídos en un CD-ROM adjunto
· Conector USB Tipo B, compatible con USB1.1/2.0
· Alimentación suministrada a través del cable USB
· 1 slot, con cubierta
· LEDs de alimentación y de acceso incluídos
· Compatibilidad comprobada con los cartuchos de juego de Konami provistos del chip de sonido SCC.
· La unidad consta de un único slot físico, pero a través de la conmutación por soft se pueden emular dos slots.
· No se garantiza el funcionamiento a través de un hub USB, por lo que rogamos que conecten el aparato directamente al puerto USB del ordenador.

3- Algunas especificaciones referidas al diseño de la unidad pueden sufrir variaciones respecto a la información contenida en la página web.
· El lector de cartuchos MSX es un periférico desarrollado para utilizar los cartuchos MSX en entorno Windows. Aquellas personas que no dispongan de cartuchos de juego MSX no podrán emplear este aparato.
· Se necesita una máquina provista de conector USB y sistema operativo Windows (98/98SE/Me/2000/XP). No funciona en Linux ni en Mac OS.
· Para ejecutar los cartuchos de juego MSX mediante este aparato se necesita la versión de MSXPLAYer distribuida conjuntamente con el segundo número de MSX Magazine.
· No se suministra el cable USB. Hay que adquirir el cable (de tipo USB B – USB A) por separado.
· El lector de cartuchos no es compatible con generadores de sonido FM, modems y demás cartuchos de expansión. Tampoco se garantiza la plena compatibilidad con todos los cartuchos de juego MSX, en particular con los fabricados en el extranjero.
· Se puede dar el caso de que un mal estado de conservación de los cartuchos impida su correcta ejecución.

4- [Acerca del MSXPC 20th Anniversary Model]
· El ordenador, con el teclado integrado en la unidad principal, viene acompañado de un ratón. Sin embargo, para utilizarlo se necesita una unidad de visualización (monitor) que debe adquirirse aparte.
· El MSXPC 20th Anniversary Model no cuenta con ningún slot para cartuchos MSX de serie. Para disfrutar con los juegos en dicho formato debe adquirirse aparte el lector de cartuchos MSX Game Reader.
· Con el ordenador no se suministra ningún paquete de software de aplicación.

5- Acerca de reparaciones y sustitución de productos defectuosos
El periodo de garantía se extenderá por el periodo de 1 año en el caso del [MSXPC 20th Anniversary Model] y de 6 meses en el caso del [MSX Game Reader], a contar a partir de la fecha de entrega. En caso de detectarse alguna anomalía durante los ocho días siguientes a la recepción del producto, se procederá a la sustitución de éste una vez comprobada su naturaleza defectuosa. A partir de este plazo, se procederá a la reparación del producto. En ambos casos se necesita el justificante de entrega, por lo que conviene conservarlo juntamente con el certificado de garantía.

Pág. 7
1- Acerca de la cancelación de pedidos

2- Sintiéndolo mucho, dadas las características especiales de estos productos no es posible efectuar cancelaciones una vez confirmada la solicitud de pedido. Sin embargo, es posible solicitar cancelaciones sobre reservas no confirmadas; de todos modos advertimos que esto puede comportar una penalización económica a determinar según la fecha en que se hubiera efectuado la reserva. Para cursar una solicitud de cancelación debe enviarse un mensaje de correo electrónico a la dirección cs@ascii-store.com con los siguientes datos:
· Número de reserva
· Nombre, dirección y número de teléfono
· Modelo y número de unidades a cancelar

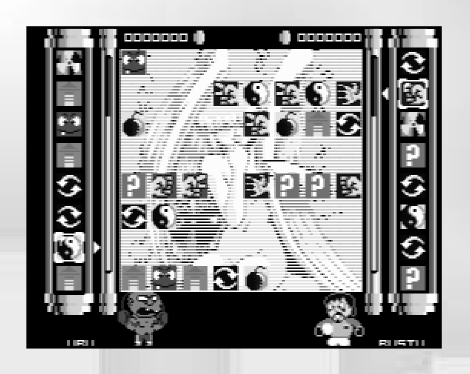
3- Las solicitudes de cancelación recibidas antes del 31/1/2004 no conllevarán gastos de cancelación, que sí que serán aplicados a las solicitudes recibidas con posterioridad a esa fecha. La cantidad no está determinada, pero sería la equivalente a los gastos generados para el trámite de la cancelación.

Pol Roca



YUPIPATI by Paxanga Soft

Este es el tercer juego de Paxanga Soft, aún en desarrollo. En esta ocasión, y sin apartarse del todo del estilo puzzle de las dos producciones anteriores, el juego esta mas orientado a competir contra un segundo jugador y, todo hay que decirlo, putearlo. Este segundo jugador puede ser tu padre, tu abuela, la vecina de al lado, tu peor enemigo, el butanero... o bien la maquina.

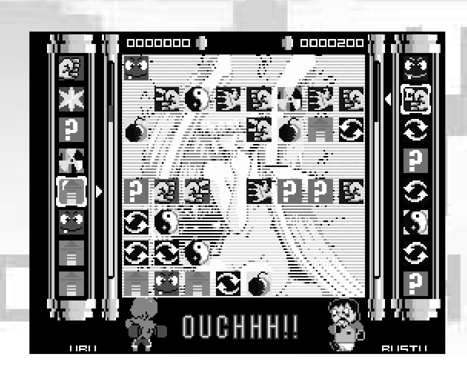


Este será el ultimo juego de puzzle de Paxanga Soft durante un buen tiempo! Vendrán mas juegos, pero de otro tipo (al menos esa es la intención).

El juego requerirá MSX2 con 128ks de RAM y 128ks de VRAM, y Msx Music para las músicas. Los efectos de sonido son en PSG. Todo ello, programado en NestorBasic y Basic Kun.

El juego

YUPIPATI se basa en el Piedra Papel o Tijera (Jan Ken Pon). De hecho YUPIPATI viene de YU-ltimeit (Ultimate) PI-edra PA-pel o TI-jera.



Contaremos con 16 personajes y una vez elegido tu luchador favorito deberás enfrentarte al resto.



Para ello dispondremos de un tablero de 8x8 casillas en las que iremos colocando una serie de fichas. Nuestro contrincante hará lo mismo, por el otro lado. Cuando una ficha lanzada por un jugador tope con la lanzada por el otro jugador (hombre o maquina), pueden pasar básicamente dos cosas

-Que gane una de las dos fichas (por ej. Piedra gana a Tijera)
-Que no gane ninguna (por ej. Papel empata con Papel)

Hasta ahí fácil, pero luego se complica algo mas cuando entran en juego ciertas fichas especiales, algunas de uso inmediato y otras que no. La cuestión es jorobar lo mas posible a tu rival. El objetivo final de todo esto es hacer llegar las fichas al otro extremo, el de tu oponente, con lo cual su barra de energía irá disminuyendo. Ya podéis imaginar que pasa cuando su barra de energía (o la tuya!) llega a cero.
Se requieren ciertas dosis de estrategia, pero tantas o mas de mala ostia.

Y todo esto viene desarrollado en tres modos de juego:

Versus, que es un combate contra otro jugador o la maquina.
Challenge, que es luchar contra todos los personajes, uno tras otro.
Tournament, que se trata de un torneo por eliminatorias, para decidir al campeón.

A día de hoy

En el momento de escribir estas líneas, YUPIPATI es casi jugable, aun cuando faltan bastantes cosas por terminar. La mayoría de personajes y gráficos del juego están terminados y las músicas, salvo que se añada alguna mas, terminadas. Falta por hacer la I.A., mejorar ciertos gráficos, el modo Torneo, las carátulas de los diskettes, manual...... la tira vaya. De hecho debería haber estado terminado para diciembre pasado pero seguramente se alargara hasta diciembre (o noviembre) próximo debido a numerosas razones extra-obsoletas mayormente.
Sin embargo es mejor así, ya que cada día se añaden nuevas cosas (que en principio no estaban planeadas) para que el juego sea mejor.
Por encima de todo se pretende dar prioridad a la jugabilidad y hacer hincapié en los piques contra el oponente.

Y esto es básicamente todo. Iré poniendo los progresos en mi pagina web siempre que me sea posible y no me sienta demasiado vago.

http://es.geocities.com/magapeich/msx.htm

Como siempre, se aceptan opiniones y comentarios. Espero que para cuando leais estas líneas, los que hayais asistido a la RU hayais podido ver la demo en vivo y en directo.

Saludos!

MsxKun



